UNIVERSIDAD POLITÉCNICA DE MADRID
FACULTAD DE INFORMÁTICA

PROBLEMAS GEOMÉTRICOS DE LOCALIZACIÓN

TESIS DOCTORAL

AYMÉE CALATAYUD RAMOS

2004

DEPARTAMENTO DE MATEMÁTICA APLICADA
FACULTAD DE INFORMÁTICA

PROBLEMAS GEOMÉTRICOS DE LOCALIZACIÓN

Autora: **Aymée Calatayud Ramos**

Directores: **Manuel Abellanas Oar**
Jesús García López de Lacalle

2004

Tribunal nombrado por el Mgfco. y Excmo. Sr. Rector de la Universidad Politécnica de Madrid, el día ..... de ..... de 2004.

Presidente D. .....

Vocal D. .....

Vocal D. .....

Vocal D. .....

Secretario D. .....

Realizó el acto de defensa y lectura de la Tesis el día .....

de ..... de .....

en .....

Calificación : .....

EL PRESIDENTE LOS VOCALES

EL SECRETARIO

Resumen

En esta memoria estudiamos problemas geométricos relacionados con la Localización de Servicios. La Localización de Servicios trata de la ubicación de uno o más recursos (radares, almacenes, pozos exploradores de petróleo, etc) de manera tal que se optimicen ciertos objetivos (servir al mayor número de usuarios posibles, minimizar el coste de transporte, evitar la contaminación de poblaciones cercanas, etc). La resolución de este tipo de problemas de la vida real da lugar a problemas geométricos muy interesantes.

En el planteamiento geométrico de muchos de estos problemas los usuarios potenciales del servicio son representados por puntos mientras que los servicios están representados por la figura geométrica que mejor se adapta al servicio prestado: un anillo para el caso de radares, antenas de radio y televisión, aspersores, etc, una cuña si el servicio que se quiere prestar es de iluminación, por ejemplo, etc. Estas son precisamente las figuras geométricas con las que hemos trabajado. En nuestro caso el servicio será sólo uno y el planteamiento formal del problema es como sigue: dado un anillo o una cuña de tamaño fijo y un conjunto de n puntos en el plano, hallar cuál tiene que ser la posición del mismo para que se cubra la mayor cantidad de puntos.

Para resolver estos problemas hemos utilizado arreglos de curvas en el plano. Los arreglos son una estructura geométrica bien conocida y estudiada dentro de la Geometría Computacional. Nosotros nos hemos centrado en los arreglos de curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo dos puntos, ya que estos fueron los arreglos con los que hemos tenido que tratar para la resolución de los problemas. De entre las diferentes técnicas para la construcción

de arreglos hemos estudiado el método incremental, ya que conduce a algoritmos que son en general más sencillos desde el punto de vista de la codificación.

Como resultado de este estudio hemos obtenido nuevas cotas que mejoran la complejidad del tiempo de construcción de estos arreglos con algoritmos incrementales. La nueva cota $O(n\,\lambda_3(n))$ supone una mejora respecto a la cota conocida hasta el momento: $O(n\,\lambda_4(n))$. También hemos visto que en ciertas condiciones estos arreglos pueden construirse en tiempo $O(n\,\lambda_2(n))$, que es la cota óptima para la construcción de estos arreglos. Restringiendo el estudio a curvas específicas, hemos obtenido que los arreglos de n circunferencias de k radios diferentes pueden construirse en tiempo $O(n^2 \cdot min(log(k), \alpha(n)))$, resultado válido también para arreglos de elipses, parábolas o hipérbolas de tamaños diferentes cuando las figuras son todas isotéticas.

Abstract

In this work some geometric problems related with facility location are studied. Facility location deals with location of one or more facilities (radars, stores, oil wells, etc.) in such way that some objective functions are to be optimized (to cover the maximum number of users, to minimize the cost of transportation, to avoid pollution in the nearby cities, etc.). These kind of real world problems give rise to very interesting geometrical problems.

In the geometric version of many of these problems, users are represented as points while facilities are represented as different geometric objects depending on the shape of the corresponding facility: an annulus in the case of radars, radio or TV antennas, agricultural spraying devices, etc. A wedge in many illumination or surveillance applications. These two shapes are the geometric figures considered in this Thesis. The formal setting of the problem is the following: Given an annulus or a wedge of fixed size and a set of n points in the plane, locate the best position for the annulus or the wedge so that it covers as many points as possible.

Those problems are solved by using arrangements of curves in the plane. Arrangements are a well known geometric structure. Here one deals with arrangements of unbounded Jordan curves which intersect each other in at most two points. Among the different techniques for computing arrangements, incremental method is used because it is easier for implementations.

New time complexity upper bounds has been obtained in this Thesis for the construction of such arrangements by means of incremental algorithms. New upper bound is $O(n\,\lambda_3(n))$ which improves the best known up to now ($O(n\,\lambda_4(n))$). It is

shown also that sometimes this arrangements can be constructed in $O(n\,\lambda_2(n))$, which is the optimal bound for constructing these arrangements. With respect to specific type of curves, one gives an $O(n^2 \cdot min(log(k), \alpha(n)))$ algorithm that constructs the arrangement of a set of n circles of k different radii. This algorithm is also valid for ellipses parabolas or hyperbolas of k different sizes when all of them are isothetic.

Índice General

Capítulo 1

Introducción

La Geometría Computacional, al menos con este nombre y con el enfoque actual, es una disciplina extremadamente nueva. Surge a finales de los años 70 a partir del área de desarrollo y análisis de algoritmos, y desde ese momento crece y se desarrolla como una disciplina importante dentro de la Ciencia de la Computación. Cuenta cada vez más con seminarios, conferencias y talleres, así como con una gran comunidad de investigadores. El éxito como área de investigación se explica parcialmente por el atractivo de los problemas propuestos y de las soluciones obtenidas.

El propósito de la Geometría Computacional es el estudio de problemas geométricos desde un punto de vista algorítmico; entendiéndose por algoritmo la secuencia finita de pasos que se han de seguir para resolver un problema. El objetivo es solucionar los problemas utilizando el menor número posible de pasos elementales de modo que aumente la eficiencia en el cálculo de la solución.

Por otro lado, la Geometría Computacional constituye una herramienta fundamental en diversas áreas de la Computación que necesitan de un enfoque geométrico, como la Computación Gráfica, la Robótica, los Sistemas de Información Geográficos, la Optimización Combinatoria, el Procesamiento de Imágenes y la Investigación Operativa, entre otras.

El término moderno que se utiliza para referirse a la Investigación Operativa es el de Teoría de Optimización. La Teoría de Optimización está relacionada con la localización eficiente de recursos que son escasos. Si nos centramos en la localización de servicios, los problemas que nos encontramos tienen que ver con la localización de uno o más servicios de manera que se optimice un determinado objetivo, como puede ser el minimizar el coste del transporte, el llegar hasta el mayor número de usuarios posibles o el ofrecer un servicio "equitativo" a los usuarios del servicio. Las aplicaciones de la Localización de Servicios son numerosas; hallar el emplazamiento de una central de bomberos, de un hospital, de una planta nuclear o de un radar son ejemplos clásicos. Otras aplicaciones las encontramos en la localización de "servicios" menos obvios, como componentes electrónicos, alarmas, aspersores, etcétera. En general, la teoría de Localización de Servicios puede ser aplicada en cualquier escenario donde se necesite hallar la mejor ubicación de un objeto, cualquiera que sea éste.

El propósito fundamental de esta tesis es estudiar problemas de localización en los que se quiere hallar la posición en el plano de una figura geométrica (el servicio) de manera que, dados un conjunto de puntos en el plano (los "usuarios" del servicio), la figura cubra la mayor cantidad de puntos. Las figuras con las que trabajamos son el anillo y la cuña. En ambos casos el tamaño de la figura es fijo.

El anillo se define como la región que se encuentra entre dos circunferencias concéntricas de radios diferentes. En la vida real el anillo es la forma que tienen muchos componentes de equipos. En los sistemas de regadío, es el área alcanzada por el agua lanzada por un aspersor. La región de cobertura de algunas antenas es así mismo un anillo de dimensiones fijadas por la potencia de emisión y la altura de la antena.

La cuña o sector es una parte del plano comprendida entre dos rayos que parten de un punto común llamado vértice de la cuña. Los rayos dividen el plano en dos cuñas, una de ángulo menor que 180 grados y otra de ángulo mayor. En la vida real las cuñas aparecen en muchos problemas de iluminación y vigilancia. Es frecuente que un foco o una cámara de vigilancia fija abarquen un sector de

amplitud fija. El problema básico en este caso consiste en situar el foco en un sitio y con una orientación tal que se maximice el número de puntos iluminados de un conjunto de puntos dado.

Para la resolución de estos problemas nos hemos valido de arreglos de curvas. Los arreglos son una estructura bien conocida por todos aquellos que se dedican a la Geometría Computacional. Sin embargo, con el propósito de que esta tesis sea autocontenida, la memoria comienza con un capítulo llamado "Técnicas de Implementación" en el que hacemos un repaso de los arreglos y otros conceptos ampliamente conocidos que van a utilizarse a lo largo de esta tesis de modo que en lo sucesivo podamos concentrarnos exclusivamente en los aspectos nuevos de los problemas a tratar.

Los arreglos con los que trabajamos son arreglos de curvas de Jordan no acotadas. En unos casos las curvas se intersectan dos a dos en a lo sumo un punto y en otros, en a lo sumo dos puntos. Para curvas que se intersectan dos a dos en a lo sumo un punto se demostró entre los años 1985 y 1986 que el arreglo de las mismas puede construirse en tiempo cuadrático utilizando un algoritmo incremental. Los algoritmos incrementales son aquellos en los que se trabaja de elemento en elemento y son muy apreciados debido a la sencillez de los pasos a realizar. La complejidad del tiempo de construcción de los arreglos se obtiene de la complejidad de las zonas de sus curvas. Lo que se demostró en realidad entre 1985 y 1986 fue que para estos arreglos la complejidad de la zona de una curva es lineal respecto al número de curvas en el arreglo. El resultado fue probado primero por Chazelle, Guibas y Lee en 1985 [1] y más tarde por Edelsbrunner, O'Rourke y Seidel en 1986 [7].

Para curvas que se intersectan dos a dos en a lo sumo dos puntos, la construcción del arreglo de n curvas mediante un algoritmo incremental consume tiempo superior a cuadrático: $O(2^{\alpha(n)}\, n^2)$. Esto se debe a que la complejidad de las zonas de las curvas en estos arreglos no es lineal sino $O(2^{\alpha(n)}\, n)$. Este resultado es un caso particular de un resultado más general que nos dice lo siguiente:

Dado un conjunto Γ de n curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo k puntos, sea γ_0 otra curva de Jordan no acotada que se intersecta con cada curva de Γ en a lo sumo un número constante de puntos. La complejidad de la zona de γ_0 en el arreglo de las curvas de Γ es $O(\lambda_{k+2}(n))$.

Cuando $k = 2$, la complejidad que se obtiene para la zona es $O(\lambda_4(n))$, siendo $\lambda_4(n) = \Theta(2^{\alpha(n)}\, n)$. La función $\lambda_s(n)$ mide la longitud máxima de una s-secuencia de Davenport-Schinzel de al menos n símbolos. Una secuencia de Davenport-Schinzel es una secuencia finita de símbolos donde no hay repeticiones inmediatas ni subsecuencias de longitud $s + 2$ donde dos símbolos alternen. Este concepto tiene su origen en un problema geométrico estudiado por Davenport y Schinzel hacia mediados de los años 60. Un poco más adelante Davenport y Schinzel proponen en un artículo el problema de estimar la longitud máxima de 3-secuencias de n símbolos. En ese mismo artículo ellos dan cotas de $O(n\, log(n))$ pero el problema de determinar si la longitud era ésta o $O(n)$ queda abierto hasta 1986 en que Hart y Sharir prueban que la longitud de estas secuencias es $O(n\, \alpha(n))$, siendo $\alpha(n)$, la inversa de la función de Ackermann, una función que crece hasta infinito, aunque de manera extremadamente lenta. Y ya a finales de los años 80 se encontraron cotas superiores e inferiores para la longitud máxima de s-secuencias de Davenport-Schinzel de longitud n. Las mejores estimaciones para $\lambda_s(n)$ son entonces las siguientes:

$\lambda_1(n) = n$,
$\lambda_2(n) = 2n - 1$,
$\lambda_3(n) = \Theta(n\, \alpha(n))$ (Hart y Sharir [6]),
$\lambda_4(n) = \Theta(n\, 2^{\alpha(n)})$ (Agarwal, Sharir y Shor [10]),
$\lambda_{2s}(n) = O(n\, 2^{O(\alpha^{s-1}(n))})$ (Agarwal, Sharir y Shor [10]),
$\lambda_{2s+1}(n) = O(n\, \alpha(n)^{O(\alpha^{s-1}(n))})$ (Agarwal, Sharir y Shor [10]).

Volvamos a los arreglos de curvas. Cuando las curvas se intersectan dos a dos en a lo sumo dos puntos, el algoritmo incremental nos lleva a construir el arreglo de n curvas en tiempo $O(n\, \lambda_4(n))$, ya que $\lambda_4(n)$ es la complejidad de la zona de una curva en el arreglo, al ser $k = 2$. Nosotros hemos probado que en este caso, o sea,

cuando $k = 2$, el arreglo puede construirse en tiempo $O(n\,\lambda_3(n))$ utilizando igualmente un algoritmo incremental. El capítulo de "Nuevas Cotas para Arreglos de Curvas" está dedicado a probar este resultado. Esta tesis está estructurada de la siguiente manera:

Capítulo 2: Técnicas de Implementación

Repaso de una serie de estructuras de datos, de algoritmos y resultados básicos que son ampliamente conocidos pero que, dado el uso reiterado que se hace de los mismos, se ha considerado obligada su presentación. Recordaremos así la definición de una transformación Dual, la función de Ackermann, los detalles de la construcción de arreglos utilizando el método incremental, el tratamiento a ciertos casos degenerados en los arreglos y la construcción de arreglos a partir de listas ordenadas de puntos de intersección. También repasamos la obtención de las curvas de nivel dentro de un arreglo.

Capítulo 3: Nuevas Cotas para Arreglos de Curvas

En este capítulo presentamos ciertos resultados nuevos que son utilizados en el resto de los capítulos. En particular, tratamos el caso de la construcción de arreglos de curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo dos puntos. Probamos que estos arreglos pueden construirse en tiempo $O(n\,\lambda_3(n))$, lo cual es una mejora con respecto a la cota que se tenía hasta el momento: $O(n\,\lambda_4(n))$.

Capítulo 4: El Problema del Anillo

En este capítulo se estudia el siguiente problema: dado un conjunto de n puntos en el plano y un anillo $\mathcal{R}$ de tamaño fijo, se quiere hallar la posición del anillo en el plano donde se cubre la mayor cantidad de puntos. Este problema lo resolvemos a través de un problema equivalente: dados un conjunto de n anillos de tamaño fijo, hallar la región del plano cubierta por la mayor cantidad de anillos. Para resolver esta versión del problema construimos el arreglo de anillos. Al considerar

el arreglo de anillos como un arreglo de circunferencias tenemos entonces que los objetos con los que trabajamos se intersectan dos a dos en a lo sumo dos puntos. Convertimos entonces cada circunferencia en dos curvas no acotadas. Utilizando los resultados obtenidos en el capítulo de "Nuevas Cotas para Arreglos de Curvas" llegamos a que el arreglo de $2n$ circunferencias puede construirse en tiempo $O(n^2)$.

Capítulo 5: El Problema de la Cuña

En este capítulo tratamos el problema de hallar la posición en el plano de una cuña de ángulo fijo cuyo vértice pertenece a un segmento también dado de manera tal que, dado un conjunto de n puntos en el plano, la cuña cubra la mayor cantidad de puntos. Para resolver este problema presentamos tres algoritmos. El primero es el más sencillo de los tres. Las únicas estructuras de datos que se utilizan son dos vectores, uno de tamaño lineal y otro de tamaño cuadrático y el único preprocesamiento que se hace es el de ordenar conjuntos de números. La complejidad de esta primera versión para hallar la solución es $O(n^2 \log(n))$. El segundo algoritmo puede considerarse también sencillo: lo único que hay que hacer es construir ciertos conjuntos e intersectarlos. La implementación, sin embargo, va a ser más trabajosa ya que será preciso construir arreglos de curvas. La complejidad baja en este caso hasta $O(n^2\,\alpha(n))$. El tercer el algoritmo es el que halla la solución en tiempo $O(n^2)$. Esta versión incluye pasar a trabajar en el problema Dual, además de construir arreglos de curvas y de insertar curvas en arreglos a los que no pertenecen.

Capítulo 2

Técnicas de Implementación

2.1 Introducción

Durante el desarrollo de la parte principal de este texto se da por sentado que se está familiarizado con todos los conceptos algorítmicos a los que se hace referencia. Se trata de conceptos básicos en los que no nos detenemos, a fin de facilitar la presentación de las nuevas estrategias.

Queremos, sin embargo, dedicar este capítulo al repaso de todas las cuestiones básicas que vamos a utilizar. Veremos así los detalles de la mezcla ordenada de listas, de cómo hallar las curvas de nivel en un arreglo, de la construcción de arreglos utilizando el método incremental y de cómo obtener un arreglo de curvas teniendo solamente el orden en que cada curva se intersecta con las demás. Comenzamos con la introducción de las estructuras de datos.

2.2 Estructuras de Datos

Arreglos de Curvas

Sea Γ un conjunto de n curvas en el plano. La definición de *arreglo inducido por el conjunto* Γ podemos encontrarla en muchos libros y artículos. A continuación citamos la definición que aparece en el libro de Sharir y Agarwal ([13], pág. 115). En la figura 2.1 vemos los elementos a que hace referencia esta definición.

Definición 2.2.1. *El arreglo* $\mathcal{A}(\Gamma)$ *se define como la partición del plano inducida por las curvas de* Γ. *Se llaman vértices a los puntos de intersección entre las curvas; aristas, a los máximos tramos conexos de las curvas que no contienen ningún vértice y caras, a las componentes conexas de* $\mathbb{R}^2 - \Gamma$.

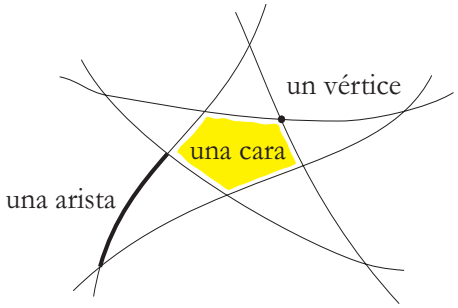


Figura 2.1: Los elementos del arreglo.

La estructura de datos que hemos escogido para representar un arreglo cualquiera de n curvas es la "winged-edge" ([9], pág. 215). En esta estructura, los datos sobre las relaciones locales de incidencia entre los diferentes elementos se almacenan explícitamente. La implementación se hace habitualmente utilizando vectores, aunque también es posible utilizar listas sin que ninguna operación sobre la estructura vea incrementada su complejidad. En este texto veremos la "winged-edge" suponiendo que se trabaja con vectores.

Para representar un arreglo se necesitan tres vectores que van a almacenar respectivamente los vértices, las aristas y las caras del arreglo. Al ser el vector una estructura estática, todos los vectores del arreglo van a ser de tamaño cuadrático, ya que es cuadrático el tamaño máximo de cualquiera de los conjuntos que almacenan. Esto quiere decir también que en algún lugar se tiene que almacenar el número de elementos que cada vector tiene. Este otro lugar puede ser alguna entrada del propio vector o una nueva estructura en la que se guardaría el tamaño de cada vector. Utilizaremos la última variante. Para ver los diferentes vectores

comenzamos por el de vértices, en el que cada entrada tiene dos campos:

Vector de Vértices: campos de la entrada

- coordenadas del vértice.
- una de las aristas incidentes en el vértice.

El vector de aristas es el que más campos tiene por entrada: once. Si llamamos e a una arista cualquiera del arreglo la información que se guarda en los campos es la que aparece a continuación.

Vector de Aristas: campos de la entrada

- vértice de partida v_p.
- vértice de llegada v_g.
- curva a la que pertenece e.
- siguiente arista incidente en v_p en sentido horario.
- siguiente arista incidente en v_p en sentido antihorario.
- siguiente arista incidente en v_g en sentido horario.
- siguiente arista incidente en v_g en sentido antihorario.
- arista que precede a e en su curva soporte.
- arista que sigue a e en su curva soporte.
- cara a la derecha de e, orientando e de v_p a v_g.
- cara a la izquierda de e, orientando e de v_p a v_g.

Por último, en el vector de caras cada entrada tiene solamente dos campos:

Vector de Caras: campos de la entrada

- una de las aristas pertenecientes a la cara.
- posición de la cara con respecto a esta arista.

Conjuntos de Curvas

Para almacenar n curvas simples utilizaremos un vector de tamaño lineal en el que cada entrada contiene tres campos. Uno de estos campos se refiere a un arreglo, y es que en nuestros problemas, los conjuntos de curvas tienen siempre un arreglo.

Vector de Curvas: campos de la entrada

- parámetros que definen la curva.
- arista del arreglo contenida en la curva.
- lista ordenada de las intersecciones de la curva en el arreglo.

El tercer campo, que se refiere a la lista de intersecciones, puede obtenerse del arreglo de curvas pero lo hemos puesto como otro campo para facilitar el trabajo posterior con las curvas.

En los algoritmos que hemos diseñado sucede a menudo que el arreglo de curvas que se quiere construir no se obtiene de una vez sino que el conjunto de curvas es dividido primero en varios conjuntos. En este caso, para cada conjunto se construye el arreglo correspondiente. Cuando utilizamos este tipo de estrategia nos interesa luego obtener el orden en que una curva de uno de los conjuntos intersecta las curvas de otro. Esto nos lleva a modificar las estructuras de datos. Ahora, en cada entrada del vector de curvas vamos a almacenar las listas de sus intersecciones con cada arreglo. Tendremos así tantas listas como arreglos haya. La entrada del vector de curvas queda en este caso con los siguientes campos:

Vector de Curvas: campos de la entrada (cuando hay más de un arreglo)

- parámetros que definen la curva.
- arreglo al que la curva pertenece.
- lista ordenada de las intersecciones de la curva con el arreglo $A[1]$.
- lista ordenada de las intersecciones de la curva con el arreglo $A[2]$.
- ...
- lista ordenada de las intersecciones de la curva con el arreglo $A[k]$.

El valor k es el número de conjuntos en que es dividido el conjunto incial de curvas y es un valor que se conoce al diseñar el algoritmo y, por lo tanto, al implementarlo. De ahí que se puede saber de antemano el número de campos que tendrán las entradas de este vector.

Los diferentes arreglos estarán almacenados en otro vector que podríamos llamar el vector de arreglos. Las entradas de este vector tendrán un solo campo:

Vector de Arreglos: campos de la entrada

- lista de los vectores que componen el arreglo (de la entrada en cuestión).

Listas de Intersecciones

Para almacenar las intersecciones de una curva con otras curvas se utilizarán listas ordenadas, en las que cada elemento de la lista guarda un punto de intersección. En los casos en que la curva sea cerrada, se asumirá que la lista es circular y que los puntos aparecen en ella en el mismo orden en que aparecen sobre la curva al recorrerla en sentido horario. La información que almacena cada elemento de la lista es la siguiente:

Campos de los elementos de la lista de intersecciones

- coordenadas del punto de intersección q.
- curva con la que se obtuvo q.
- entero que nos indica si q es el único punto de intersección entre las curvas.
- en caso de ser el único punto de intersección, entero que nos indica si es un punto de tangencia o no.
- en caso de no ser el único punto de intersección, entero que nos indica si es el de más a la izquierda o el de más abajo (en caso de que las dos intersecciones tengan igual abscisa).

Del cuarto campo se deduce que las curvas pueden ser tangentes entre sí. Los tres últimos campos son necesarios a la hora de construir el arreglo de curvas a partir de las listas, como veremos en §2.7.

2.3 La Función de Ackermann

En esta sección vamos a recordar la definición de la función de Ackermann ya que su inversa, la función $\alpha(n)$, aparece en las cotas de muchos de los resultados obtenidos.

La función de Ackermann $A(n)$ es una función definida en el conjunto de los números naturales de la siguiente manera. Primeramente se definen las funciones que tenemos a continuación, donde $A_k^{(n)}$ es la notación utilizada para la composición de funciones. Así, $A_k^{(n)}$ significa la composición $A \circ A \circ \ldots \circ A$ de A con ella misma n veces.

$$\begin{aligned} A_1(n) &= 2n, \\ A_k(n) &= A_{k-1}^{(n)}(1), \quad k \geq 2, \end{aligned}$$

Luego se hace

$$A(n) = A_n(n).$$

Para trabajar un poco con la recursividad en la definición de las $A_k(n)$ veamos que

$$\begin{aligned} A_2(n) = A_1^{(n)}(1) &= A_1(A_1(\ldots(A_1(1))\ldots)) \\ &= A_1(A_1(\ldots(2)\ldots)) \\ &= 2^n \end{aligned}$$

$$\begin{aligned} A_3(n) = A_2^{(n)}(1) &= A_2(A_2(\ldots(A_2(1))\ldots)) \\ &= A_2(A_2(\ldots(2)\ldots)) \\ &= 2^{2^{2^{\cdot^{\cdot^{\cdot^2}}}}} \quad (n \text{ números } 2) \end{aligned}$$

Al hacer $A(n) = A_n(n)$ se tiene que $A(1) = 2$, $A(2) = 4$, $A(3) = 16$ y $A(4)$ es una torre exponencial, como la de $A_3(n)$ pero con 65536 números 2. De este modo, la función de Ackermann es una función que crece extremadamente rápido. Para ver las propiedades básicas de las funciones hasta ahora vistas váyase al artículo de Tarjan [15].

Las funciones inversas de A_k y A se denotan por α_k y α respectivamente. De este modo, sus definiciones son las siguientes: para todo $n \in \mathbb{N}$,

$$\begin{aligned} \alpha_1(n) &= \lceil \tfrac{n}{2} \rceil, \\ \alpha_2(n) &= \lceil \log(n) \rceil, \\ \alpha_k(n) &= min\{s \geq 1 : \alpha_{k-1}^s(n) = 1\}. \end{aligned}$$

Es decir, $\alpha_k(n)$ es el número de iteraciones que α_{k-1} tiene que hacer para ir de n a 1. En particular, la función $\alpha_3(n)$ se denota comúnmente por $\log^*(n)$ y es igual al menor número de veces que hay que hacer logaritmos para ir desde n hasta 1. Visto de otra manera, $\alpha_3(n)$ es la menor cantidad de números 2 que hay que utilizar en la torre $2^{2^{\cdot^{\cdot^{\cdot^2}}}}$ para que $2^{2^{\cdot^{\cdot^{\cdot^2}}}}$ sea mayor o igual a n. Todas las funciones $\log(n)$ (en A_k y α_k) son en base 2. La función $\alpha(n)$ se define entonces como sigue:

$$\alpha(n) = min\{k \geq 1 : A(k) \geq n\}.$$

La función $\alpha(n)$, al igual que todas las funciones $\alpha_k(n)$, es una función no decreciente que tiende a infinito cuando el argumento tiende a infinito. Las funciones $\alpha_k(n)$ crecen de manera extremadamente lenta, y esto se cumple también para la función $\alpha(n)$, que crece más lentamente que cualquiera de las $\alpha_k(n)$. En particular, $\alpha(n) \leq 4$ para todo $n \leq A(4)$, que es el número formado por 65536 números 2 en una torre exponencial; de modo que $\alpha(n) \leq 4$ para todos los valores "imaginables" de n.

2.4 Mezcla de Listas Ordenadas

Lema 2.4.1. *La mezcla ordenada de las listas ordenadas $l_1, l_2, \ldots, l_k$ puede hacerse en tiempo $O(n \log_2(k))$, siendo $n = |l_1| + |l_2| + \ldots + |l_k|$.*

Demostración. La mezcla se hace por pasos. En cada paso se agrupan las listas por pares y se mezclan, obteniendo $\lceil \frac{k}{2} \rceil$ nuevas listas ordenadas, con las que se va al siguiente paso y se repite el procedimiento. Terminamos cuando se obtiene una única lista. Al tratarse de k listas, el número de pasos será $\lceil log_2(k) \rceil$. Como hay n elementos distribuidos en las listas, cada paso requiere tiempo lineal. La complejidad total viene dada entonces por el producto del número de pasos por la complejidad de cada paso: $O(n \log(k))$. □

2.5 Curvas de Nivel

Dado un arreglo $\mathcal{A}(\Gamma)$ de n curvas monótonas no acotadas, la curva de nivel i se define como el conjunto de todas las aristas y vértices de $\mathcal{A}(\Gamma)$ que tienen estrictamente i curvas de Γ por encima (o por debajo, según sea más cómodo para el problema que se esté tratando). Denotaremos por c_i a la curva de nivel i. Dada esta definición, la curva de nivel cero es la que coincide con la envolvente superior de $\mathcal{A}(\Gamma)$.

Hallar las curvas de nivel significa en este caso etiquetar los vértices y las aristas de $\mathcal{A}(\Gamma)$ con el nivel al que pertenecen. Podemos además establecer las adyacencias entre estos elementos de modo que podamos luego recorrer una curva de nivel yendo de arista en arista.

En los problemas que tratamos, sin embargo, no necesitaremos nunca recorrer una curva de nivel, por lo que no veremos los campos necesarios para esto. Aunque las caras no pertenezcan a las curvas de nivel, las incluiremos dentro del etiquetado, porque este dato sí que nos hará falta más tarde. Decimos entonces que la cara $\mathcal{F}$ de $\mathcal{A}(\Gamma)$ tiene nivel i si hay i curvas de Γ por encima de $\mathcal{F}$. Esto quiere decir que en las entradas del vector de aristas, el de vértices y el de caras, incluiremos un campo más para el nivel. En la siguiente figura vemos todos los elementos que en el arreglo del dibujo tienen nivel dos.

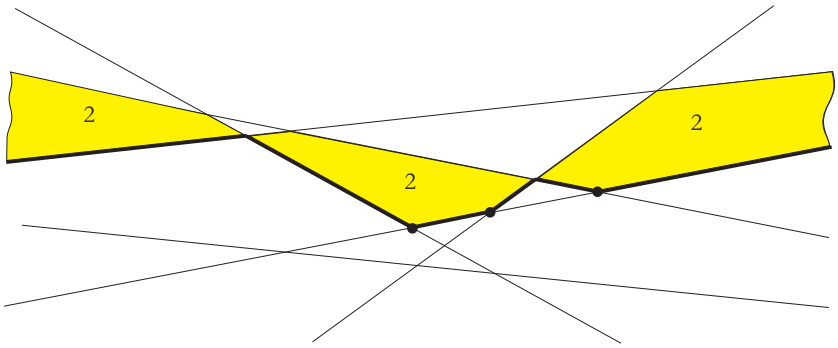


Figura 2.2: Caras, aristas y vértices de nivel dos.

De la definición de curva de nivel se deduce que cualquier vertical intersecta en a lo sumo un punto cualquier curva de nivel y que, además, los puntos de intersección de las diferentes curvas de nivel aparecen ordenados en la vertical. Sea l una recta vertical cualquiera. Insertemos l en el arreglo. Sean x_0, x_1, ..., x_{n-1} los puntos de intersección entre $\mathcal{A}$ y l ordenados en sentido decreciente de sus ordenadas. Tenemos entonces que $x_0 \in c_0$, $x_1 \in c_1$, ..., $x_{n-1} \in c_{n-1}$, siendo c_0, c_1, ..., c_{n-1} las curvas de nivel desde la cero hasta la $n-1$. Para hallar las diferentes curvas de nivel partimos entonces del único punto que conocemos de cada curva: los diferentes puntos x_i. Veremos primeramente cómo hallar la curva de nivel cero.

El punto x_0 puede pertenecer a una arista o a un vértice de $\mathcal{A}$. Como en la curva de nivel cero, y en cualquier otra, es más fácil pasar de una arista a otra de su mismo nivel que hacerlo desde un vértice, haremos que el punto x_0 pertenezca a una arista. Para esto, hacemos que l esté a la izquierda del mínimo de las abscisas de los vértices de $\mathcal{A}$. De esta manera, todos los puntos x_i van a pertenecer a aristas. Llamaremos entonces e_i a la arista soporte de x_i.

Otra consecuencia de posicionar l de esta manera es que el resto de las aristas de un determinado nivel i estarán todas hacia la derecha de e_i. Esto quiere decir también que en las caras de nivel i las aristas de su mismo nivel irán apareciendo ordenadas en sentido antihorario en la frontera de la cara.

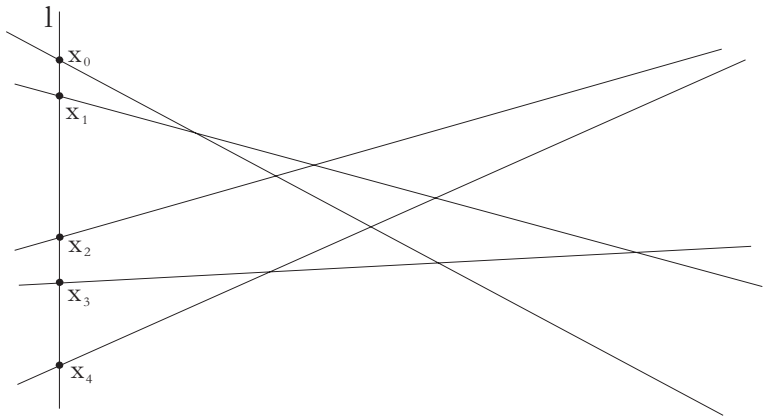


Figura 2.3: La recta l a la izquierda de todos los vértices de $\mathcal{A}$.

Volvamos a la curva de nivel cero que queremos hallar. A partir de e_0 tenemos que encontrar las demás aristas que pertenecen a c_0. En el caso que nos ocupa, por coincidir c_0 con la envolvente superior, tendremos sólo una cara de nivel cero. Las aristas y vértices de nivel cero serán todas las que pertenecen a la frontera de esta cara.

Sea $\mathcal{F}$ la cara que está a la izquierda de e_0 al recorrer e_0 de izquierda a derecha. Lo primero que hacemos es etiquetar $\mathcal{F}$, e_0 y los vértices extremos de e_0 con nivel

cero. El resto de las aristas y vértices de nivel cero aparecen al bordear $\mathcal{F}$, para lo cual nos valemos de los campos de la arista que nos dan las aristas adyacentes dentro de una misma cara. De esta manera ya tenemos c_0.

Sea ahora x_i otro cualquiera de los puntos de intersección de l, siendo $i \neq 0$. Después de haber etiquetado e_i y sus vértices extremos con nivel i buscamos la cara $\mathcal{F}$ que queda a la izquierda de e_i al recorrer e_i de izquierda a derecha y le damos también nivel i. Luego comenzamos a bordear $\mathcal{F}$ en sentido antihorario etiquetando las demás aristas y vértices que van apareciendo hasta llegar a un vértice v que tenga ya etiqueta. En estos vértices es donde se pasa de una cara a otra del mismo nivel.

En la figura 2.4 tenemos un ejemplo. La curva c_1 que aparece destacada y las caras $\mathcal{F}$, $\mathcal{G}$ y $\mathcal{H}$ tienen nivel 1. Los vértices u y v son los vértices donde pasamos de una cara de nivel 1 a otra también de nivel 1 cuando recorremos c_1 de un extremo a otro. Tanto u como v tienen nivel 0.

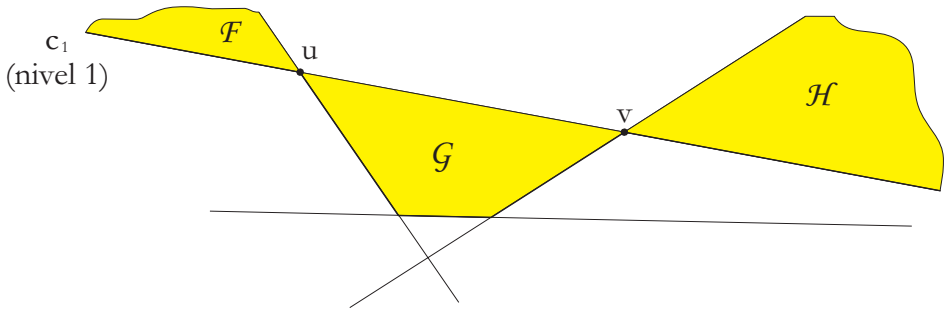


Figura 2.4: Las caras de nivel 1 y los vértices de nivel 0.

Para dar con la siguiente cara de un mismo nivel tenemos que seleccionar, de todas las aristas incidentes en u, cuál es la que pertenece a c_i. Una vez que tengamos esta arista se repite el proceso: la etiquetamos, buscamos la cara que queda a su izquierda, la etiquetamos y la bordeamos en sentido antihorario etiquetando todas las aristas y vértices que aparecen hasta llegar a otro vértice que tenga ya etiqueta. El proceso termina cuando lleguemos a una arista que se prolongue hasta el infinito, o cuando comprobemos que la cara bordeada es la cara exterior.

Como vemos, los vértices que hayan sido etiquetados por una curva de nivel anterior van a mantener su etiqueta. Esto se deduce de la definición de curva de nivel y del modo en que las hallamos. Luego, si por un vértice pasan varias curvas de nivel, el nivel de éste será el de la primera curva que lo encuentre.

Sea v un vértice que ha sido ya etiquetado por una curva de nivel anterior a la curva c_i que nos ocupa en un determinado momento. Para evitar consumir tiempo en seleccionar la próxima arista $e^{'}$ de nivel i de entre todas las incidentes en v vamos a añadir a los vértices otro campo: *siguiente*. En este campo vamos a guardar, cada vez que etiquetemos un vértice v con nivel j, la arista que está más a la derecha de las dos de nivel j que son incidentes en v. Dadas las aristas e y $e^{'}$ de nivel j, decimos que $e^{'}$ es la de más a la derecha si aparece después de e durante el etiquetado de c_j. Luego, al llegar a v, la próxima arista de nivel i será, de todas las incidentes en v, la adyacente a *siguiente* en sentido horario. Véase que si por un vértice v pasan varias curvas de nivel y, por cada par de aristas que tienen el mismo nivel y son incidentes en v consideramos la que está más a la derecha tenemos que éstas aparecen alrededor de v ordenadas en sentido horario según su nivel.

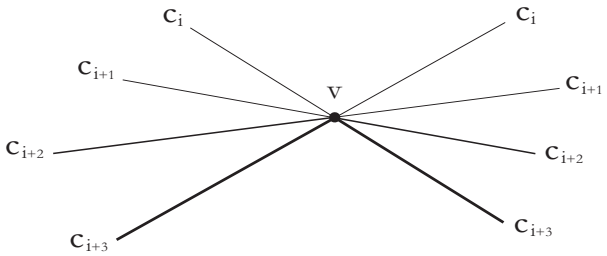


Figura 2.5: Las aristas ordenadas alrededor de v.

Esto quiere decir que teniendo la arista *siguiente* asociada a v podemos encontrar siempre $e^{'}$ en tiempo constante gracias al campo de las aristas que nos da la siguiente (anterior) en sentido horario (antihorario). De esta manera, el procesamiento que se hace en los vértices ya etiquetados consume tiempo constante.

En el resto de los vértices no se hace ningún procesamiento, solamente son etiquetados.

Teorema 2.5.1. *Etiquetar los elementos de un arreglo como* $\mathcal{A}(\Gamma)$ *con el nivel al que pertenecen puede hacerse en tiempo* $O(n^2)$.

Demostración. Localizar una de las aristas en cada curva de nivel se hace en tiempo lineal. En este caso se ha localizado la arista de más a la izquierda en cada curva de nivel. Desde cada una de estas aristas se comienza el recorrido por la curva de nivel. Durante el recorrido se etiquetan las aristas, caras y vértices que vayan apareciendo. Cada arista del arreglo es visitada a lo sumo una vez y el trabajo que se hace en cada vértice lleva tiempo constante. □

2.6 Algoritmo Incremental

Sea Γ un conjunto de n curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo un punto. En esta sección recordaremos la estrategia para la construcción del arreglo $\mathcal{A}(\Gamma)$ utilizando el método incremental.

Como es sabido, con el método incremental las curvas se añaden al arreglo de una en una. El arreglo es inicialmente una estructura vacía y se actualiza en cada inserción. La complejidad del tiempo de construcción de $\mathcal{A}(\Gamma)$ es $O(n^2)$.

Sea γ una curva de Jordan no acotada que intersecta cada curva de Γ en a lo sumo un punto. La zona de γ en el arreglo $\mathcal{A}(\Gamma)$ se define como el conjunto de caras del arreglo que tienen intersección con γ y se denota por $z(\gamma)$.

Veremos primero el caso de los arreglos conexos y luego, los arreglos no conexos - la nueva estructura de datos para los mismos y la complejidad del tiempo de construcción.

2.6.1 Arreglos Conexos

El mecanismo para insertar una curva γ en un arreglo de otras n curvas es a grandes rasgos el siguiente [11]. En tiempo lineal se obtiene una curva δ que tenga intersección con γ. Dada la estructura de datos con la que contamos, a través de δ tenemos acceso en tiempo constante a una de sus aristas dentro del arreglo. En este momento podemos recorrer δ en tiempo lineal hasta encontrar la arista e que se intersecta con γ, pues desde una arista cualquiera del arreglo se podrá ir siempre en tiempo constante a las dos que le son adyacentes dentro de la misma curva. Una vez hallada e, pasamos a las dos caras que separa. Ambas caras van a pertenecer a $z(\gamma)$.

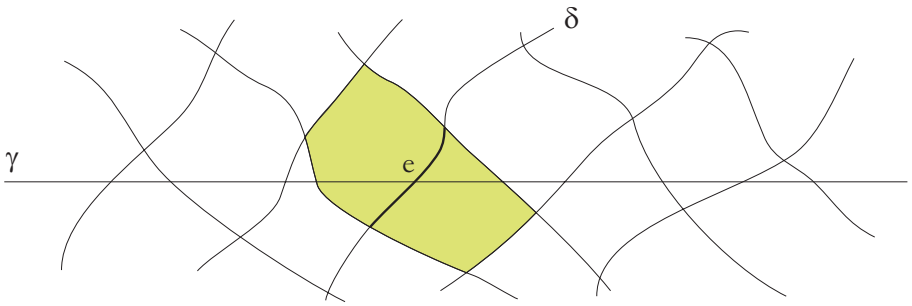


Figura 2.6: La arista e y sus dos caras adyacentes.

En cada una de las caras se recorren entonces las aristas de la frontera. Para esto, cada arista fue almacenada con información que permite el acceso a las dos aristas que le son adyacentes dentro de la misma cara. El objetivo es obtener, en cada cara, todas las aristas que tienen intersección con γ. De estas aristas se escoge entonces la que da acceso a la próxima cara que γ visita. De esta manera se va pasando por todas las caras de la zona de γ. El orden en que las caras de $z(\gamma)$ son visitadas nos permite dibujar el recorrido de γ dentro del arreglo.

La actualización de la estructura de datos es como sigue. Supongamos que hemos entrado a una cara $\mathcal{F}$ por el punto q y que, después de haber hallado todas las demás intersecciones entre la frontera de $\mathcal{F}$ y γ vemos que el punto q' es el que está a continuación de q en γ. Vamos al vector de vértices y en la siguiente en-

trada libre almacenamos el punto q'. Vamos al vector de aristas y hacemos lo mismo con la nueva arista qq'. Sean ahora a y b los vértices de la arista de $\mathcal{F}$ que contiene a q'. La arista ab ha quedado así dividida en dos: aq' y $q'b$. Esto quiere decir que hay que añadir al vector de aristas otra arista más, la aq', por ejemplo, y actualizar ab para convertirla en $q'b$. La actualización incluye cambiar el extremo a por q' y establecer las adyacencias entre aristas en el nuevo extremo q'. Una vez que tenemos los vectores de vértices y aristas construimos el vector de caras haciendo un recorrido del arreglo primero en profundidad. El trabajo con el vector de caras lo tenemos en §2.7.

2.6.2 Arreglos no Conexos

Al trabajar con arreglos no conexos, una de las opciones para tratarlo es ver la posibilidad de convertir el arreglo en uno conexo, bien sea mediante la adición de una nueva curva que tenga intersección con cada una de las componentes conexas, bien sea mediante la prolongación de las curvas iniciales. En ocasiones, sin embargo, no tendremos más remedio que trabajar con el arreglo en su versión no conexa. En esta sección vamos a ver la representación de los arreglos no conexos y el análisis de la complejidad de construcción de los mismos cuando se utiliza un algoritmo incremental.

Antes de ver la estructura vamos a tener en cuenta que insertar una curva γ en el arreglo requiere localizar primero las componentes conexas con las que tiene intersección y luego, saber que todas estas componentes conexas pasarán a ser una después de insertar γ. Por su parte, la unión de varias componentes conexas en una sola requiere hallar primero el orden en que éstas aparecen a lo largo de γ.

Unir varias componentes en una significa formar un nuevo vector de vértices, uno de aristas y otro de caras a partir de la unión de los respectivos vectores de vértices, aristas y caras de las componentes de partida. Para facilitar las operaciones de unión de componentes conexas utilizaremos listas y no vectores para la representación de estos arreglos. El motivo es que las listas nos permiten hacer la unión de dos componentes en tiempo constante, al único coste de tener, además

de un apuntador al primer elemento de la lista, otro al último elemento.

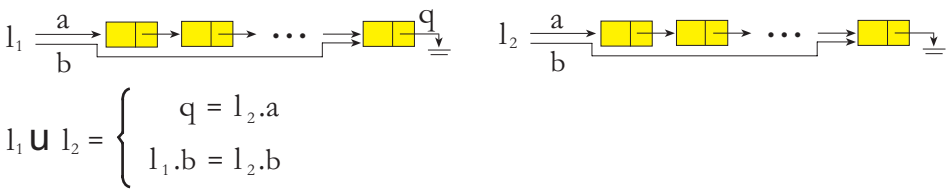


Figura 2.7: La unión de dos listas.

Si se utilizaran vectores, la unión de dos componentes no sería posible en tiempo constante ya que sería preciso escribir uno por uno los elementos de un vector en el otro. Aunque la representación de los conjuntos de vértices, aristas y caras se haga a través de listas nos referiremos a estas listas como vectores.

Además de los vectores de vértices, aristas y caras, los subarreglos (componentes conexas) tendrán un nuevo vector: el vector de curvas, donde se almacenarán las curvas que intervienen en la componente conexa en cuestión. El conjunto de componentes conexas conforman el arreglo como tal. Finalmente, un arreglo no conexo no va a ser otra cosa que una lista de componentes conexas. En la siguiente figura tenemos un esquema de esta estructura.

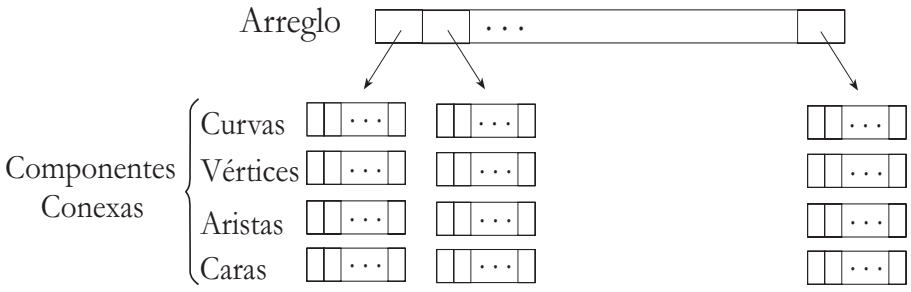


Figura 2.8: La estructura para almacenar un arreglo no conexo.

Los pasos para insertar una curva γ en un arreglo no conexo son entonces los siguientes:

[1] En cada componente conexa, visitar las curvas que la forman hasta encontrar una que tenga intersección con γ, en cuyo caso insertamos γ y marcamos la componente como "modificada".

[2] Crear una nueva componente conexa igual a la unión de las componentes modificadas.

[3] Eliminar las componentes modificadas.

Veremos ahora que siguiendo los pasos anteriores la complejidad del tiempo de construcción de un arreglo con varias componentes conexas es la misma que la complejidad del tiempo de construcción del arreglo si éste fuera conexo. Veremos que la complejidad viene dominada por el tiempo de inserción de las curvas en los subarreglos. A este tiempo lo denotaremos por $t(n)$, siendo n el número de curvas que hay en la componente conexa en cuestión cuando γ es insertada. De este modo $t(n)$ puede ser $O(n)$, $O(n\,\alpha(n))$, $O(n\,2^{\alpha(n)})$ o cualquier otra función que cumpla $f(n_1) + f(n_2) + \ldots + f(n_k) \leq f(n)$ siendo $n_1 + n_2 + \ldots + n_k \leq n$, o sea, para cualquier función convexa. Haremos el análisis suponiendo que $t(n) = O(n)$ (para los demás casos se comprueba que se llega a las mismas conclusiones).

Paso [1]: sea n_i el número de curvas que hay en la $i-$*ésima* componente conexa. Visitar todos las curvas de una componente i y luego insertar γ cuesta $O(n_i)$. La complejidad del primer paso es a lo sumo lineal, al ser $n_1 + n_2 + \ldots + n_k \leq n$, siendo k el número de componentes conexas del arreglo.

Paso [2]: la unión de componentes conexas trae consigo dos operaciones. Una es construir los vectores de la nueva componente conexa y la otra, añadir al vector de aristas las nuevas aristas que aparecen: aquellas que unen entre sí las componentes conexas en cuestión. Cada uno de los vectores de la nueva componente conexa se construye haciendo la unión de los respectivos vectores de las componentes conexas modificadas. En el nuevo vector de curvas tenemos que cuidar que la curva que se acaba de insertar aparezca solamente una vez. Como dos vectores se unen en tiempo constante, ya que en realidad se trata de listas,

los vectores de k componentes conexas pueden unirse en tiempo a lo sumo lineal, por cumplirse $k \leq n$. En la segunda operación, si se trata de unir k componentes, habrá que añadir a lo sumo k nuevas aristas. Una vez que se conozcan los extremos de las aristas, esta operación tiene entonces complejidad a lo sumo lineal. Sin embargo, para saber cuáles son los extremos de las aristas y actualizar correctamente las adyacencias entre éstas y las que ya existían, hay que determinar primero el orden en que las nuevas aristas aparecen a lo largo de γ. Veremos que hallar este orden para cada uno de las curvas que se inserta tiene complejidad final $O(n \log(n))$.

Sea k_i el número de componentes conexas que hay en el paso i. El valor $k_i - k_{i+1} + 1$ nos da el número de componentes conexas que tuvieron intersección con γ en el paso i, que no es otra cosa que el número de componentes conexas que se unieron y pasaron a ser una sola en el paso i. Véase que cuando la curva γ que se inserta pertenece a una sola componente conexa el valor anterior nos da uno. Hallar el orden en que las componentes conexas aparecen a lo largo de γ cuesta entonces $O((k_i - k_{i+1} + 1) \log(k_i - k_{i+1} + 1))$ en el paso i. Luego de insertar n curvas el tiempo consumido ordenando habrá sido

$$(k_1 - k_2 + 1)\ \log(k_1 - k_2 + 1)\ +\ (k_2 - k_3 + 1)\ \log(k_2 - k_3 + 1)\ + \ldots +$$
$$+\ (k_{n-1} - k_n + 1)\ \log(k_{n-1} - k_n + 1)$$

Dado que en ningún paso se pueden unir más de n componentes conexas se cumple que $k_i - k_{i+1} + 1 \leq n$ para todo i, con lo que se tiene la siguiente acotación:

$$(k_1 - k_2 + 1)\ \log(k_1 - k_2 + 1)\ +\ (k_2 - k_3 + 1)\ \log(k_2 - k_3 + 1)\ + \ldots\ +$$
$$+\ (k_{n-1} - k_n + 1)\ \log(k_{n-1} - k_n + 1)\ \leq$$
$$(k_1 - k_2 + 1)\, log(n)\ +\ (k_2 - k_3 + 1)\, log(n)\ +\ \ldots\ +$$
$$+(k_{n-1} - k_n + 1)\, log(n)\ \leq$$
$$\{\ (k_1 - k_2 + 1)\ +\ (k_2 - k_3 + 1)\ + \ldots +\ (k_{n-1} - k_n + 1)\ \}\, log(n)\ =$$

$$= \ (k_1 - k_n + n - 1)\, log(n) \ \leq \ 2n\, log(n)$$

La complejidad del segundo paso es $O(n \log n)$ en total.

Paso [3]: el tercer paso se hace en tiempo a lo sumo lineal para cada inserción, ya que se trata solamente de eliminar componentes conexas. La complejidad total de este paso es entonces $O(n^2)$.

Por todo lo visto, la complejidad del tiempo de construcción de estos arreglos viene determinada por el paso [1], que es el paso donde se insertan las curvas en el arreglo. Luego, al igual que ocurre en los arreglos conexos, el tiempo de construcción de los arreglos no conexos depende del tiempo de inserción de las curvas en los mismos.

2.7 Algoritmo Incremental. Listas de Puntos

La construcción de un arreglo a partir de listas ordenadas de puntos de intersección va a aparecer como uno de los pasos en varios de los algoritmos que presentamos. La razón es que en algunos casos, si se construye el arreglo partiendo de las curvas como tal y utilizando el método incremental, no está asegurado que la complejidad final sea cuadrática.

Sin embargo, hallando primero el orden en que cada curva se intersecta con los demás podemos construir el arreglo de modo que todo el tiempo consumido sea cuadrático: esto quiere decir que tanto el orden en que cada curva se intersecta con las demás como la construcción del arreglo partiendo de este orden pueden hacerse en tiempo cuadrático.

En esta sección vamos a ver la construcción incremental de un arreglo partiendo, no de las curvas como tal, sino de sus respectivas listas ordenadas de puntos de intersección. Veremos además que la complejidad del tiempo de construcción es $O(n^2)$.

Como ya sabemos, los puntos de intersección de una curva se encuentran almacenados en una lista ordenada. Dado que nuestros arreglos van a ser de curvas que se intersectan dos a dos en a lo sumo dos puntos, este será el caso para el que veremos los detalles. Supondremos que por un punto de intersección pasan a lo sumo dos curvas. Las curvas conque trabajamos pueden ser tangentes entre sí.

Sean $\{\gamma_1,\ \gamma_2,\ \ldots,\ \gamma_n\}$ el conjunto de curvas en el orden en que aparecen en el vector de curvas. Las curvas se insertarán en el arreglo en este orden: el del vector de curvas. Insertar una curva significa recorrer su lista asociada e ir añadiendo al arreglo las nuevas aristas y vértices. Todo elemento de la lista se va a transformar en un vértice del arreglo y todo par de elementos consecutivos de la lista va a dar lugar a una arista. En el caso de curvas cerradas, la última arista a insertar será la que enlaza al último elemento de la lista con el primero. El vector de caras lo construimos al final, después que hayamos insertado todas las curvas.

Como un punto de intersección pertenece a dos curvas, podemos encontrarnos conque, a la hora de tratarlo, éste se encuentre ya en el arreglo, insertado por una curva anterior, en cuyo caso no podemos volver a insertarlo. Ver el ejemplo de la figura 2.9.

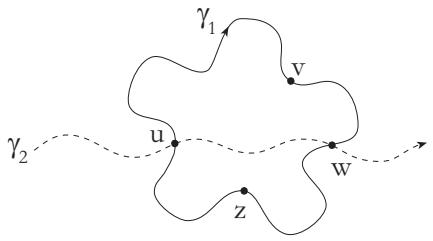


Figura 2.9: γ_1 ha insertado vértices de curvas que aún no han aparecido.

En este ejemplo, cuando se trabaje con γ_2 nos encontraremos conque los vértices u y w ya están en el arreglo, insertados por γ_1. Detectar si un vértice está en arreglo o no se hace comparando los subíndices de las curvas en el vector de curvas ya que las curvas se insertan siguiendo el orden que guardan en el vector de

curvas. Por otra parte, las dos curvas que pasan por un vértice se obtienen de la lista en la que el vértice aparece.

La segunda implicación de encontrarnos un punto de intersección que está ya en el arreglo como un vértice v es que, aunque no lo insertemos, sí tenemos que insertar la arista que de él parte en la curva que estamos tratando, ya que una arista sí pertenece a una única curva. En este caso tenemos que saber, a partir del punto de intersección y de la lista que estamos tratando, cuál es el vértice que lo representa en el vector de vértices, de entre todos los vértices que están ya en el arreglo (en el vector de vértices), de modo que podamos enlazar correctamente la nueva arista y actualizar las que ya están en el arreglo y son adyacentes a v.

Para tal fin vamos a cambiar el vector de vértices por una matriz de vértices. La matriz será de dos dimensiones. Este cambio se hace sobre la base de que dos curvas se intersectan en a lo sumo dos puntos y de que en una matriz podemos ordenar mejor los vértices.

La Matriz de Vértices

La matriz de vértices será una matriz de tamaño $n \times n$ a la que llamaremos $\mathcal{M}$. Tanto las filas como las columnas van a corresponder a las n curvas. De esta manera, los dos posibles vértices a que dan lugar dos curvas γ_i y γ_j se guardarán en las entradas $\mathcal{M}[i,j]$ y $\mathcal{M}[j,i]$ de la matriz. Las posiciones de la diagonal, por lo tanto, no tendrán sentido en esta matriz, pues ninguna curva se intersecta con ella misma. Luego, antes de que el arreglo esté construido, sabemos ya dónde se va a almacenar cada punto de intersección. Las entradas de la matriz tendrán los mismos campos que tenían las entradas del vector de vértices. La información está organizada del siguiente modo dentro de $\mathcal{M}$. Sea v un vértice donde se intersectan γ_i y γ_j. Hacemos:

$\mathcal{M}[i,j] = v$, siendo $i > j$, si γ_i y γ_j se intersectan en un sólo punto, en este caso v, o si γ_i y γ_j se intersectan en dos puntos y v es el de más a la izquierda o el de más abajo (en caso de que los dos puntos sean de igual abscisa).

$\mathcal{M}[i,j] = v$, siendo $i < j$, en caso contrario, o sea, si γ_i y γ_j se intersectan en dos puntos y v es el de más a la derecha o el de más arriba (en caso de que los dos puntos sean de igual abscisa).

La comprobación de si un punto es el único donde se intersectan dos curvas o si es el de más a la izquierda o el de más abajo puede hacerse en tiempo constante para cada par de curvas. Los cálculos pueden hacerse en el momento, cada vez que se necesite la información, o puede hacerse una única vez, en cuyo caso la información ha de guardarse en algún lugar. Nos quedaremos con la última versión. Esta información será calculada entonces mientras construimos las listas de puntos de interseción y será almacena en las listas, con cada punto de intersección. Ver §2.2.

Insertando una Curva en el Arreglo

Sean v_1, v_2, ..., v_k los puntos de intersección que están almacenados en la lista l_i asociada a la curva γ_i. Para cada punto de intersección vamos a preguntar primero si existe ya como un vértice del arreglo. La respuesta es negativa si el subíndice de la curva con la que se obtuvo dicho punto de intersección es mayor que i. En este caso tenemos que añadir un nuevo vértice al arreglo. Después, hayamos tenido que añadir el vértice o no, sí tendremos que añadir la nueva arista.

Las aristas se añaden del siguiente modo: por cada elemento de la lista añadimos la arista que de él parte y sólo para el primer elemento añadiremos además la arista que en él termina. Esto último no será necesario para el caso de curvas cerradas.

Añadiendo un Vértice al Arreglo

Si el vértice está ya en la matriz no hay que añadir nada a la matriz de vértices. Supongamos entonces que tratamos con un vértice v_m de γ_i que no está insertado aún en el arreglo. Esto quiere decir que la curva γ_j con la que se obtiene v_m está después de γ_i en el vector de curvas. El primer paso para insertar el vértice es determinar su posición dentro de $\mathcal{M}$ y el segundo, llenar los campos de esta posición.

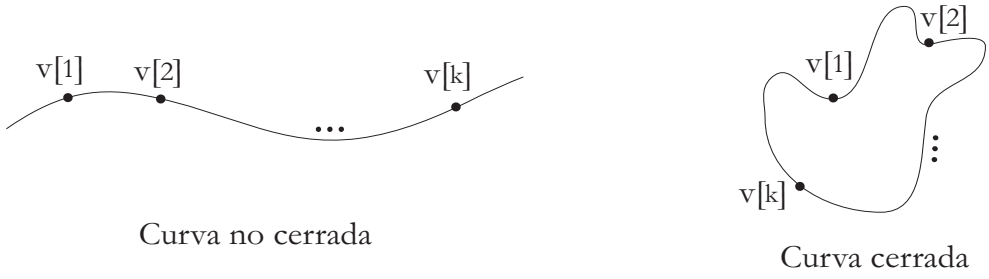


Figura 2.10: Los vértices y las aristas en ellos incidentes.

El método para determinar la posición de v dentro de $\mathcal{M}$ lo acabamos de ver en párrafos anteriores. Como bien se dijo ahí, toda la información que hace falta está almacenada con v en la lista.

Los campos a rellenar en la entrada de $\mathcal{M}$ son dos (ver §2.2). Las coordenadas las tenemos asociadas a v_m en la lista. Como arista adyacente a v_m ponemos la que parte de v_m: $v_m v_{m+1}$. Esta arista no está aún en el vector de aristas pero será inmediatamente insertada una vez que acabemos con v_m, con lo que podemos saber cuál va a ser su posición dentro del vector de aristas: la próxima entrada libre.

Añadiendo una Nueva Arista al Arreglo

Insertar la arista $v_m v_{m+1}$ en el vector de aristas significa rellenar los once campos de la entrada correspondiente, que supondremos será la número k. Los campos relativos a las caras serán rellenados más tarde, ya que el vector de caras se construye una vez que se tiene el arreglo. Sea $e = v_m v_{m+1}$. Veremos los campos en el orden en que aparecen en §2.2.

Vértices de partida y de llegada: el vértice de partida es siempre el que estamos tratando en la lista, v_m, y el de llegada, el que sigue a este dentro de la lista, v_{m+1}, a menos que se trate de la primera arista de la curva, en cuyo caso no existe el vértice de partida y v_m es el de llegada. En estos casos almacenamos en el campo correspondiente al vértice de partida un índice negativo que indique que está indefinido.

Curva a la que pertenece e : γ_i.

Siguiente arista incidente en v_m en sentido horario y antihorario: estas aristas pertenecen a la otra curva que pasa también por v_m, llamémosla γ_j (hemos supuesto que estamos insertando γ_i en el arreglo). Que la otra curva que pasa por v_m es γ_j es información que viene almacenada con v_m en la lista, así como el hecho de si se trata de curvas tangentes o no. Supondremos que no son tangentes. El caso de tangencia se hace de manera similar. Si $j > i$, quiere decir que γ_j no ha sido aún insertada en el arreglo, por lo que dejamos estos campos pendientes. Supongamos entonces que $j < i$. En este caso v_m fue insertado en el arreglo por γ_j. Por lo tanto, como arista incidente en el vértice (este es uno de los campos que tiene un vértice) tendrá una de las dos que estamos buscando. Ver figura 2.11. Una vez que tengamos esta arista podemos saber, mirando en sus campos, si v_m es su punto de partida o su punto de llegada. También podemos hallar la otra arista de γ_j que tiene también a v_m en uno de sus extremos. Llamemos entonces f y f' a las aristas de γ_j que son incidentes en v_m, siendo f la que tiene a v_m como vértice de partida.

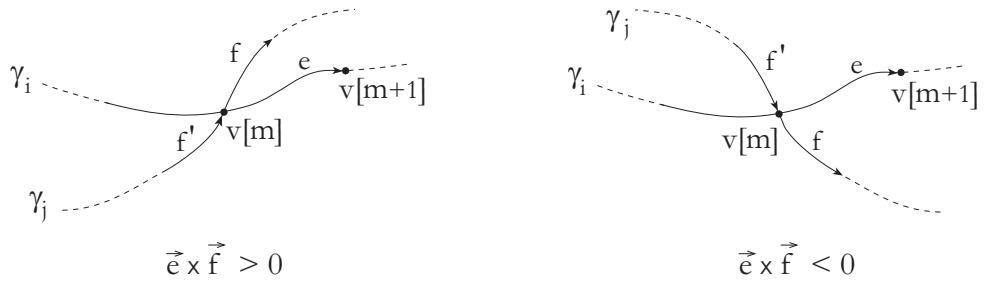


Figura 2.11: Las aristas incidentes en v_m.

Cuando hablamos de producto vectorial y utilizamos la notación $\overrightarrow{e} \times \overrightarrow{f}$ nos estamos refiriendo a los vectores tangentes a γ_i y γ_j en el punto $v[m]$. Hagamos entonces el producto vectorial $\overrightarrow{e} \times \overrightarrow{f}$. Si el resultado es positivo, f es la arista incidente en v_m en sentido antihorario y f', la incidente en v_m en sentido horario. Cuando el resultado es negativo los papeles entre f y f' se invierten y f pasa a

ser la arista incidente en v_m en sentido horario.

Una vez que hemos terminado con estos campos de e completamos estos mismos campos en f y f', ya que ahora sí existen las aristas incidentes en v_m provenientes de una curva diferente de γ_j. Por ejemplo, si $\overrightarrow{e} \times \overrightarrow{f}$ fue positivo, en la entrada de la arista f escribimos, en el campo correspondiente, que e es la arista incidente en $v[m]$ en sentido horario.

En el caso de tangencia la situación sería como la que vemos en la figura 2.12.

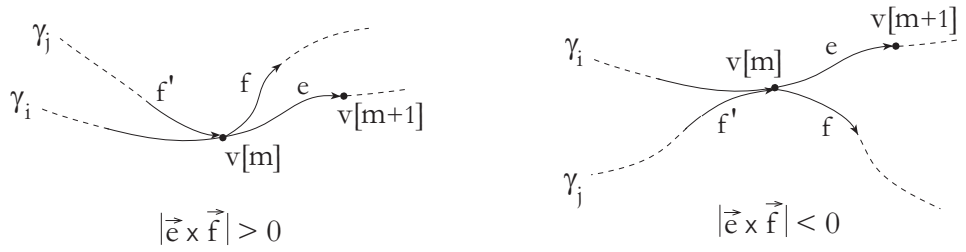


Figura 2.12: Las aristas incidentes en v_m cuando γ_i y γ_j son tangentes.

Siguiente arista incidente en v_{m+1} en sentido horario y antihorario: idem al análisis para v_m.

Arista que precede a e en su curva soporte: esta arista es la que se insertó en el vector de aristas justo antes de e. Por lo tanto, es la arista de la entrada $k-1$. En caso de tratarse de la primera arista de la curva, no existe la arista que la precede en la misma curva, por lo que damos a este campo un índice negativo que así lo indique. En el caso de curvas cerradas, la primera arista insertada por una curva sí tendrá arista que la preceda, que no será otra que la última arista que la curva inserte. Como al insertar la primera arista de la curva no conocemos aún cuál será la posición de la última dentro del vector de curvas, dejamos este campo pendiente pero guardamos en una variable *var* su posición dentro del vector, de modo que cuando insertemos por fin la última arista podamos rellenar el campo pendiente en tiempo constante.

Arista que sigue a e en su curva soporte: esta arista no existe aún en el vector de aristas pero será la que se inserte a continuación de e por lo que podemos dar su localización dentro del vector: la posición $k+1$. De la misma manera que en el campo anterior, si e es la última arista de la curva, en este campo guardamos un índice negativo, y para curvas cerradas, escribimos la posición que quedó almacenada en *var*.

Actualizando el Vector de Curvas

En el vector de curvas hay que almacenar la arista que va a permitir ir desde una curva al arreglo en que aparece. Ver este campo en §2.2. La arista a almacenar con cada curva será la primera arista de la curva que se inserta en el arreglo (o lo que es lo mismo, que se inserta en el vector de aristas). Almacenar una arista significa guardar, en el campo correspondiente, la posición que la arista ocupa dentro del vector de aristas del arreglo.

Siguiente Iteración

Al llegar a este punto ya hemos terminado con todo lo relativo al punto de intersección v_m de la lista: insertar v_m como un vértice del arreglo y $v_m v_{m+1}$ como una arista. Se pasa entonces al siguiente punto de intersección de la lista, v_{m+1}, y se hace lo mismo: insertarlo como un vértice e insertar $v_{m+1}v_{m+2}$ como una arista.

Ejemplo

Sea $D = \{D_1, D_2, \ldots, D_n\}$ un conjunto de circunferencias. Veremos a grandes rasgos el tratamiento de las listas y de la matriz $\mathcal{M}$ durante la construcción de los vectores de $\mathcal{A}(D)$. Supongamos que le toca el turno a la circunferencia D_3 cuando ya han sido insertadas en el arreglo D_1 y D_2. Como vemos en la figura del ejemplo, tanto D_1 como D_2 tienen intersección con D_3. Esto quiere decir que cuando se recorrieron las listas l_1 y l_2 de estas dos circunferencias se insertaron los vértices correspondientes a D_3: v_{31}, v_{13}, v_{32} y v_{23}. Como consecuencia, las entradas de la matriz $\mathcal{M}[3,1]$, $\mathcal{M}[1,3]$, $\mathcal{M}[3,2]$ y $\mathcal{M}[2,3]$ están actualizadas.

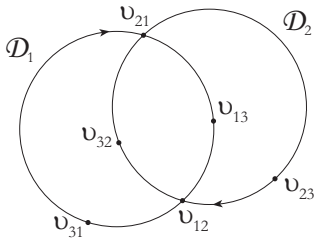


Figura 2.13: Los vértices insertados por las listas asociadas a D_1 y D_2.

Le toca el turno a D_3. Recorremos su lista asociada $l_3 = v_{31},\ v_{32}, v_{13}, v_{23}, v_{34}, v_{43}$ por ejemplo, y comprobamos que los primeros cuatro vértices están ya en el arreglo, por lo que el tratamiento de estos primeros cuatro elementos de l_3 nos lleva solamente a añadir las aristas que los unen, que serán cuatro en total: $v_{31}v_{32}$, $v_{32}v_{13}$, $v_{13}v_{23}$ y $v_{23}v_{34}$.

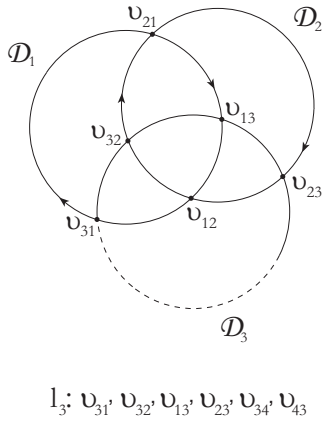


Figura 2.14: La estructura después de haber insertado los primeros cuatro elementos de D_3.

Al tratarse de curvas cerradas, con el primer vértice se añade sólo una arista, la que de él parte, y se hace $var = v_{31}v_{32}$. Cuando se termine la lista de vértices y se inserte la última arista, $v_{43}v_{31}$, se rellena su campo *arista que le sigue en su curva soporte* con el valor almacenado en *var*. Luego se va a *var* y se rellena su campo *arista que le precede en su curva soporte* con el valor de la arista que se analiza en ese momento: $v_{43}v_{31}$.

Después de pasar por los primeros cuatro vértices de la lista y ver que ya estaban le toca el turno a v_{34} y v_{43}, que no están aún en el arreglo. Estos vértices tienen como circunferencia asociada a D_4. Esta circunferencia no ha sido aún tratada, por aparecer después de D_3 en el vector de curvas, al ser $4 > 3$. Añadimos entonces v_{34} y v_{43} a $\mathcal{M}$. La lista l_3 nos dice que v_{43} es el vértice de más a la izquierda, por lo que hacemos $\mathcal{M}[4,3] = v_{43}$ y $\mathcal{M}[3,4] = v_{34}$. El siguiente y último paso es insertar las dos nuevas aristas: $v_{34}v_{43}$ y $v_{43}v_{31}$.

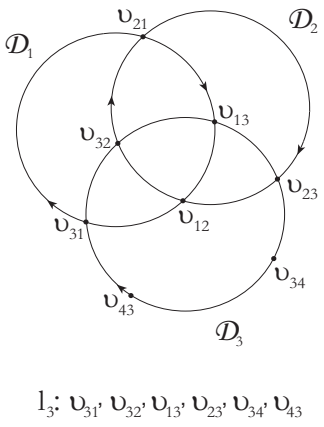


Figura 2.15: El arreglo después de insertar D_3.

El Vector de Caras

Teniendo ya los vértices y las aristas del arreglo vamos a construir el vector de caras. Paralelamente, vamos a actualizar en el vector de aristas todos los campos que quedaron pendientes: los que hacían referencia a las caras. Cuando hayamos terminado, el vector de caras tendrá todas las caras del arreglo. Asociada a cada una de las caras vendrá una de las aristas de su frontera y la posición de la cara con respecto a esta arista. En cuanto al vector de aristas, queda por almacenar cuáles son las caras que están a la izquierda y a la derecha de cada arista.

Por construcción, todas las aristas aportadas por una misma curva aparecen contiguas en el vector de aristas. La estrategia con respecto al vector de aristas será entonces la siguiente: visitar arista por arista, con lo cual estaremos recorriendo las curvas del arreglo, arista por arista, en sentido creciente de las abscisas de sus puntos de intersección, en el caso de curvas no cerradas, y en sentido horario en el caso de las curvas cerradas. Para cada arista e, nombramos la cara que tiene a su derecha: $\mathcal{H}$, por ejemplo, y en este momento recorremos esta cara por las aristas de su frontera escribiendo en cada una a qué lado tiene a $\mathcal{H}$. La posición de una cara con respecto a las aristas que la rodean se deduce del sentido en que se recorre la cara y de la orientación de las aristas. Todas las aristas orientadas en el mismo sentido que e tendrán la cara $\mathcal{H}$ a su derecha. La orientación de las aristas se deduce a su vez de sus extremos.

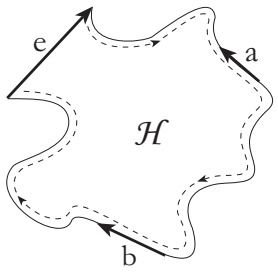


Figura 2.16: $\mathcal{H}$ está también a la derecha de b porque b y e tienen la misma orientación.

Cuando terminamos con $\mathcal{H}$ volvemos a e y creamos ahora la cara que tiene a su izquierda: $\mathcal{G}$, por ejemplo, y repetimos el proceso anterior para $\mathcal{G}$: recorrer las aristas de su frontera escribiendo en cada una hacia qué lado tienen a $\mathcal{G}$.

Al terminar con e, visitamos la arista que le sigue en el vector de aristas, f, por ejemplo, y hacemos los mismo: crear primero la cara que tiene a su derecha, visitar las aristas en la frontera de esta cara, crear la cara que tiene a su izquierda y actualizar las aristas de la frontera de esta cara. En este caso, sin embargo, tenemos que chequear primero si los campos que buscamos han sido actualizados ya o no. Por ejemplo, en caso de que ya f tenga su cara de la derecha, esta cara no se visita, y pasamos a la cara que tiene a la izquierda, que, en caso de existir,

tampoco se visita, con lo que el siguiente paso es ir a la próxima arista del vector de aristas.

En cuanto al vector de caras, cada cara nueva se almacena con una de las aristas de su frontera y con la posición de la cara con respecto a dicha arista.

Análisis de la Complejidad

Teorema 2.7.1. *Un arreglo de n curvas de Jordan no acotadas puede construirse a partir de listas de intersecciones en tiempo $O(n^2)$.*

Demostración. La complejidad del algoritmo depende del número de elementos en las listas, del tiempo consumido por elemento, y del tiempo que lleva construir el vector de caras. El número total de elementos en las listas va a ser a lo sumo $O(n^2)$ ya que este es el número de puntos de intersección al tomar las curvas de dos en dos. El tiempo que se consume por elemento es $O(1)$, que es el tiempo que se requiere para crear el vértice y la arista correspondiente.

Vértices: crear un vértice significa preguntar primero si ya existe, en cuyo caso no hay que hacer nada. Si no existe, se busca primero su entrada dentro de la matriz de vértices y acto seguido, se rellenan los campos de la entrada. Ninguna de la información a almacenar requiere de cálculos previos; toda se obtiene de la lista.

Aristas: la posición en el vector no hay que buscarla ya que las aristas se insertan una a continuación de la otra. Insertar una arista se traduce en rellenar los once campos de la entrada correspondiente. En algunos casos la información a almacenar se tiene de manera inmediata y en otros hay que calcularla, lo que puede llevarnos hasta un producto vectorial, pero en cualquier caso, el tiempo consumido es $O(1)$.

Caras: el tiempo para construir el vector de caras es $O(n^2)$ en un principio ya que la estrategia que se sigue es la de recorrer todas las aristas del vector de aristas, que son a lo sumo $O(n^2)$. Por cada arista e se mira (en tiempo $O(1)$) si

ya están en el vector de caras las dos caras que e separa, en cuyo caso rellenamos en la arista los campos relativos a la cara en cuestión. Si alguna de las dos caras no existe, se crea en ese momento, lo que conlleva a llenar en tiempo $O(1)$ una nueva entrada del vector de caras con dos datos: la arista en la que estamos y la posición de la cara con respecto a la arista. Cada vez que se añade una nueva cara al vector de caras, se recorren las aristas de su frontera. El tiempo empleado en cada una de estas aristas es $O(1)$. La frontera de cada cara se recorre sólo una vez: cuando la cara se añade al vector. Finalmente, el tiempo consumido en el vector de caras es el tiempo consumido en recorrer el vector de aristas, $O(n^2)$ en el peor de los casos, más el tiempo consumido en recorrer la frontera de cada cara en el arreglo, o sea, cada arista dos veces a lo sumo, puesto que una arista pertenece a la frontera de dos caras. Finalmente, todo el proceso equivale a recorrer tres veces el vector de aristas, y el tiempo consumido es $O(n^2)$. □

2.8 Casos Degenerados

Un arreglo de curvas es llamado simple si no existe un punto en el que se intersecten más de dos curvas. Cuando un arreglo no es simple se dice que es degenerado. Hasta ahora hemos supuesto que se trabaja siempre con arreglos simples. La razón es que los algoritmos y los teoremas son en general más sencillos cuando se tienen arreglos simples. En esta sección vamos a ver que los arreglos degenerados pueden construirse en el mismo tiempo en que se construiría su equivalente no degenerado. Con "equivalente no degenerado" queremos decir el arreglo que se obtiene al hacer una pequeña transformación en los puntos de degeneración para que por cada punto de intersección pasen solamente dos curvas.

En general tendremos menos trabajo cuando, al insertar una curva, nos encontramos con puntos de intersección que pertenecen a vértices. En la figura 2.17 tenemos un ejemplo. En este caso, q y q' son dos vértices degenerados de la curva γ que se está insertando, que es la que aparece más oscura.

Al llegar al punto de intersección q', por ejemplo, no hay que crear un nuevo vértice ni dividir en dos ninguna arista, como ocurría antes cuando la intersec-

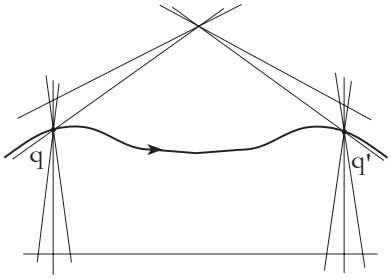


Figura 2.17: Cuando el arreglo no es simple.

ción se daba en una arista. Los únicos pasos que sí tenemos que mantener son los de almacenar la nueva arista $qq^{'}$ y actualizar las adyacencias entre esta arista y las ya existentes. La nueva tarea que aparece en estos casos es la de determinar la próxima cara visitada por la curva, de entre todas aquellas que son incidentes en el vértice degenerado. La estrategia que se sigue es la siguiente.

Sean $a_1, a_2, \ldots, a_k$ las aristas incidentes en un vértice v y γ, la curva que pasa por v. A las aristas y caras que están alrededor de v las llamaremos respectivamente aristas y caras de v. Construyamos una figura geométrica $\mathcal{K}$, de k lados, con centro en v, de modo que cada lado pertenezca a una de las caras de v y cada vértice, a una de las aristas de v.

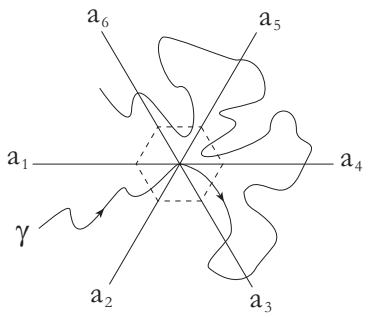


Figura 2.18: $\mathcal{K}$ es la figura dibujada con guiones.

La distancia h que existe entre v y cada uno de los vértices de $\mathcal{K}$ será constante. Para hallar el valor de h, definimos h_i del siguiente modo: si no hay intersección

entre la curva y la arista a_i, h_i es igual a la longitud de a_i, si la arista a_i tiene un solo punto de intersección con γ, h_i es igual a la distancia entre v y el punto de intersección y si a_i y γ se intersectan en más de un punto, h_i es igual a al mínimo de las distancias entre v y cada uno de los puntos de intersección. h se define entonces como la mitad del mínimo de los valores h_i. Escogiendo h de este modo, todos los lados de $\mathcal{K}$ van a tener un número par de puntos de intersección con γ, excepto aquellos que pertenecen a las caras donde γ entra y sale de v, en las que el número de puntos de intersección va a ser impar. Estas caras van a ser sólo dos y de ellas ya conocemos una, que es aquella por donde γ entra en v, puesto que es la que acabamos de dejar atrás; la otra cara, por lo tanto, será la siguiente en el recorrido de γ. Los pasos para hallar esta cara pueden resumirse del siguiente modo:

Estrategia para Vértices Degenerados

[1] Hallar las intersecciones entre γ y cada arista a_i.

[2] Hallar los valores h_i.

[3] Calcular el mínimo de los h_i, dividirlo por dos y almacenarlo en h.

[4] Calcular las intersecciones entre γ y cada lado de $\mathcal{K}$.

[5] Contar el número de intersecciones por lado y quedarnos con los dos lados que tienen un número impar.

[6] De estos dos lados, escoger el que pertenece a la cara que buscamos.

La complejidad de los pasos [1], [2] y [3] depende del número de aristas incidentes en el vértice y del número de intersecciones entre γ y una arista y la de los pasos [4] y [5], del número de intersecciones entre γ y una recta. El paso [6] se hace en tiempo constante. El tiempo consumido en v depende entonces de tres factores:

- el número de aristas incidentes,
- el número de veces que γ puede intersectar una arista a_i y
- el número de veces que γ puede intersectarse con una recta.

Como el número de aristas incidentes es el doble del número de curvas que pasan por v, podemos reescribir la última frase del párrafo anterior sustituyendo la palabra arista por la de curva. Tenemos entonces que el tiempo consumido en v depende del número de curvas incidentes, del número de veces que γ puede intersectar una curva, y del número de veces que γ puede intersectarse con una recta. Si llamamos k, s y r respectivamente a los valores anteriores, podemos calcular el tiempo consumido en v a través de la siguiente ecuación: $k + sk + rk = (1 + s + r)k$. Si r y s son cantidades constantes, el tiempo consumido en v es proporcional al número de curvas incidentes en v, que será a lo sumo una cantidad lineal. Teniendo en cuenta que el número de vértices a lo largo de γ puede llegar a ser también una cantidad lineal llegaríamos a la conclusión de que la complejidad de insertar un curva en este tipo de arreglos es cuadrática. Sin embargo, si hacemos un análisis más detallado vemos que la complejidad de inserción es lineal, siempre y cuando r y s sean constantes.

Sea γ la curva que se inserta y s el número de veces que γ puede intersectar cada curva del arreglo. Si el arreglo tiene n curvas cuando γ se inserta, el número de puntos de intersección entre γ y el arreglo será a lo sumo sn. Sean q_1, q_2, ..., q_{sn} los puntos intersección de γ y k_1, k_2, ..., k_{sn} el número de curvas incidentes respectivamente en cada uno de estos puntos. Si hacemos $c = 1 + s + r$, el tiempo empleado por el paso nuevo será en total el siguiente: $ck_1 + ck_2 + \ldots + ck_{sn} = c(k_1 + k_2 + \ldots + k_{sn})$. Como cada curva puede aparecer a lo sumo s veces a lo largo de γ se cumple entonces que $k_1 + k_2 + \ldots + k_{sn} \leq sn$; luego, $c(k_1 + k_2 + \ldots + k_{sn}) \leq csn = (1 + r + s)sn$.

Tenemos entonces que, si s y r son constantes, el tiempo total gastado por γ en los puntos de degeneración es $O(n)$. En nuestros problemas, s y r tomarán solamente los valores uno o dos. Esto quiere decir que el hecho de que el arreglo

sea degenerado no hace aumentar la complejidad del tiempo de construcción del mismo si este tiempo es $O(n^2)$ o mayor.

Las Listas y los Casos Degenerados

En las listas asociadas a curvas de arreglos simples, con cada punto de intersección de una curva γ se guarda la curva con la que se obtuvo. Ver sección §2.2.

Cuando los arreglos son degenerados, por un punto de intersección pueden pasar varias curvas. La estructura para almacenar una lista de puntos de intersecciones es ahora más general. Cada elemento de la lista va a guardar las coordenadas del punto de intersección q de γ y una lista de todas las curvas que pasan por este punto. A estas listas las llamaremos sublistas. Con cada elemento de la sublista guardamos ahora si q es el único punto de intersección entre γ y la curva en cuestión, o si hay más de uno. En nuestros problemas, "más de uno" va a significar "dos". En este último caso, se almacena si q es el punto de más a la izquierda o el de más abajo. En la figura 2.19 tenemos un ejemplo en el que las curvas son circunferencias. Vemos así la lista asociada a la circunferencia D_3 para el arreglo dado.

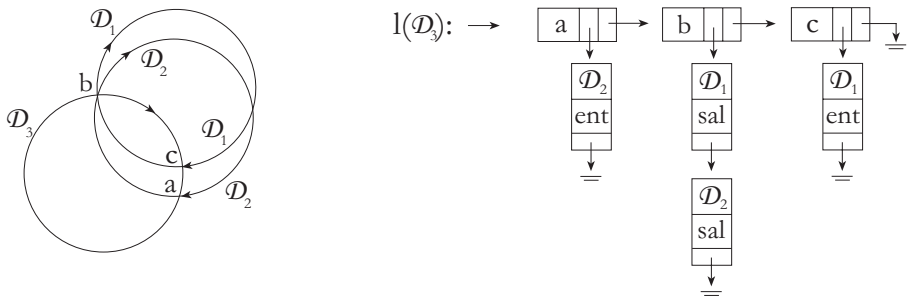


Figura 2.19: La lista de la circunferencia D_3.

En esta sección queda también incluido el caso de tangencia. Sin embargo, por el método que se usa para insertar las curvas, dado que por un punto pueden pasar ahora varias curvas, el caso de tangencia queda contemplado implícitamente. Esta es la razón por la que en esta estructura de datos, cuando el punto de intersec-

ción entre dos curvas es único, no hay que especificar si se trata de un punto de tangencia o no.

Cuando se mezclan dos listas ordenadas l_1 y l_2 y $\exists p$ tal que $p \in l_1$ y $p \in l_2$, en la lista resultante l, p aparecerá sólo una vez, y su sublista asociada va a estar formada por la concatenación de las sublistas asociadas a p en l_1 y l_2.

Arreglos Degenerados Construidos desde Listas

Ahora analizamos la complejidad del tiempo de construcción de arreglos degenerados que se obtienen a partir de listas ordenadas de puntos de intersección. El número máximo de intersecciones entre dos curvas cualesquiera del arreglo va a ser dos. Como ya sabemos, el tratamiento con una lista consiste en añadir vértices y aristas al arreglo.

Sea $\mathcal{A}$ uno de estos arreglos. Supongamos que insertamos l en el arreglo cuando ya han sido insertadas n listas. Sean $e_1, e_2, \ldots, e_j$ los elementos de l. Como el arreglo es degenerado, con cada elemento e_i viene un punto de intersección v_i y la sublista de todas las curvas que son incidentes en v_i.

Sea e_i el elemento de γ que se inserta. El trabajo con v_i no se ve afectado por el hecho de que el arreglo sea degenerado: si el vértice ya está en el arreglo no hay que hacer nada y, si no está aún, se añade. Con las aristas sí tenemos más trabajo cuando v_i está en el arreglo, ya que éste puede tener varias aristas incidentes. En este caso, para establecer las adyacencias entre la nueva arista f y las ya existentes tenemos que dar con aquellas dos entre las que se encuentra f. Para esto utilizamos la estrategia para vértices degenerados vista al inicio de esta sección. La complejidad de este paso es entonces lineal para cada objeto.

El resto de los pasos para construir el arreglo desde las listas no se ve afectado por el hecho de que el arreglo sea degenerado. Estos pasos son los de rellenar las entradas del vector de vértices, del vector de aristas y construir el vector de caras.

Teorema 2.8.1. *La construcción de arreglos degenerados partiendo de listas de puntos de intersección puede hacerse en tiempo* $O(n^2)$.

2.9 Dualidad

Utilizar la técnica del Dual para resolver un determinado problema consiste en seleccionar una transformación mediante la cual el problema dado se convierte en otro. Ciertas propiedades del problema de partida se convierten entonces en otras ciertas propiedades del problema nuevo. Al problema de partida se le llama "Problema Primal" y al obtenido mediante la transformación, "Problema Dual". Una vez que se resuelve el problema Dual, la solución se convierte en la solución del problema Primal aplicándole la transformación señalada pero ahora en sentido inverso. La transformación que se utiliza es llamada "transfomación dual". Al objeto o^* del problema Dual obtenido del objeto o del problema Primal se le llama "objeto dual".

La transformación $\mathcal{T}$ que vamos a utilizar en esta memoria es la conocida transformación que hace corresponder a un punto una recta y viceversa. En particular, a cada punto $p = (a, b)$ se le hace corresponder la recta $p^* : y = ax - b$ y de manera simétrica, toda recta $r : y = mx + n$ tiene asociado el punto $r^* = (m, -n)$. ($\mathcal{T}$ no está definida para rectas verticales). Recordemos ahora algunas de las propiedades de esta transformación dual.

Propiedad 2.9.1. $\mathcal{T}$ *preserva las relaciones de incidencia entre puntos y rectas no verticales:* $p \in r$ *si y sólo si* $r^* \in p^*$.

Consecuencia de esta propiedad es que k puntos del Problema Primal alineados según una recta r no vertical pasan a convertirse en el Dual en k rectas que pasan todas por un mismo punto, que no es otro que el transformado de r. Si construimos el arreglo de las rectas duales tenemos entonces la siguiente observación.

Observación 2.9.1. Todas las rectas no verticales que en el Problema Primal pasan por al menos dos puntos están presentes en el arreglo dual como uno de sus vértices.

Propiedad 2.9.2. *$\mathcal{T}$ preserva las relaciones de orden entre puntos y rectas.*

Esto quiere decir que en el Problema Primal el punto $p = (a, b)$ está por encima de la recta r de ecuación $y = mx + n$ si y sólo si en el Problema Dual el punto $r^* = (m, -n)$ está por encima de la recta $p^* : y = ax - b$. Decimos que un punto está por encima de una recta si ésta es intersectada por el rayo vertical que se extiende desde el punto hacia menos infinito.

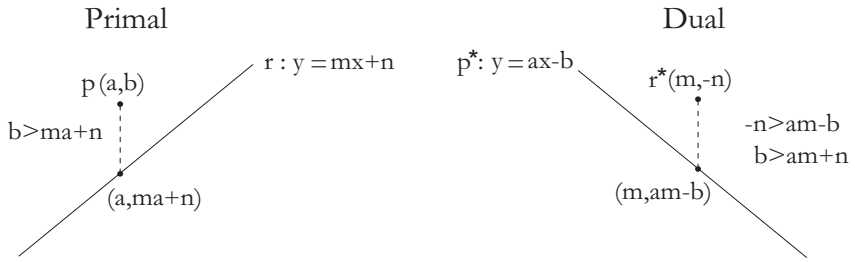


Figura 2.20: Relaciones entre el Problema Primal y el Problema Dual.

Capítulo 3

Nuevas Cotas para Arreglos de Curvas

3.1 Introducción

En este capítulo presentamos una nueva cota para el tiempo de construcción de arreglos de curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo dos puntos. Comenzamos recordando definiciones de términos que vamos a utilizar y viendo los resultados que se tienen hasta ahora sobre esta cuestión.

Definición 3.1.1. *Un* arco de Jordan *es una imagen del intervalo cerrado unitario según una aplicación continua e inyectiva.*

Definición 3.1.2. *Una* curva de Jordan no acotada *es una imagen del intervalo abierto unitario, según una aplicación continua e inyectiva, que separa el plano.*

Sea Γ un conjunto de n curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo s puntos. Sea γ_0 una curva de Jordan no acotada que se intersecta con cada una de las curvas del arreglo en a lo sumo una cantidad constante de puntos. La complejidad de la zona de γ_0 en el arreglo $\mathcal{A}(\Gamma)$ se obtiene de su relación con la complejidad de una cara en un arreglo de arcos de Jordan.

Teorema 3.1.1. *La complejidad de una cara en un arreglo de n arcos de Jordan en el plano que se intersectan dos a dos en a lo sumo s puntos es* $O(\lambda_{s+2}(n))$ *([13], pág. 117).*

Para determinar la complejidad de la zona de γ_0 a partir de este teorema lo que se hace es lo siguiente ([13], pág. 125): las curvas de Γ se cortan por sus puntos de intersección con γ_0 y se deja un pequeño espacio en estos puntos de corte. Con esta transformación, todas las caras de la zona de γ_0 pasan a formar parte de una misma cara, que pertenece ahora a un arreglo de arcos de Jordan, con el único coste de haber hecho aumentar el número de elementos del arreglo en un factor constante. En la figura 3.1 tenemos un ejemplo.

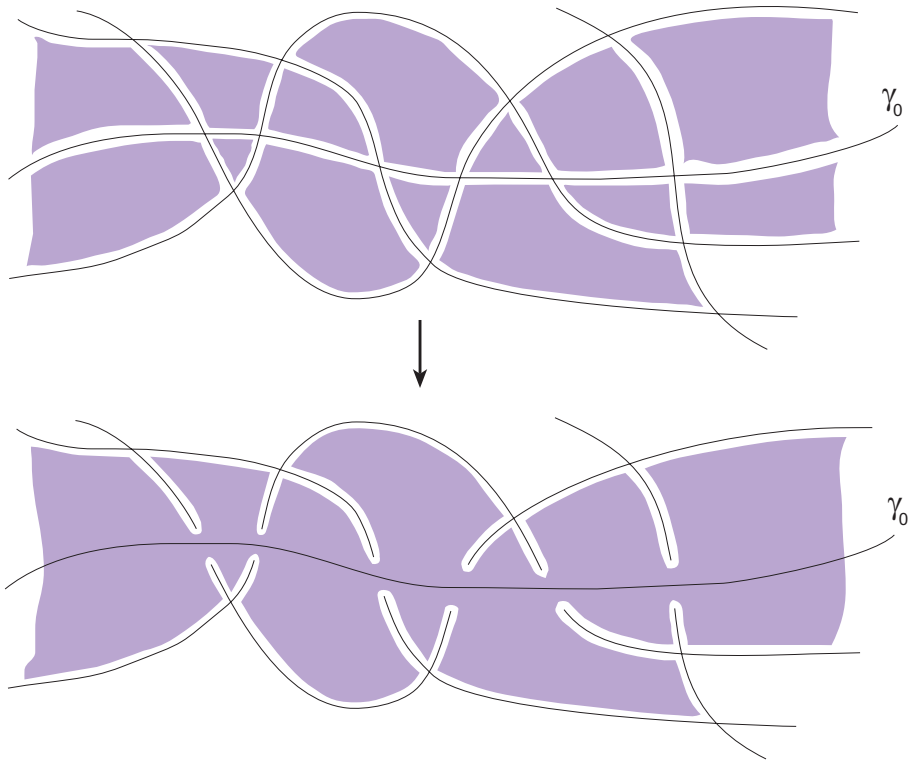


Figura 3.1: Transformación para determinar la complejidad de la zona de γ_0.

Teorema 3.1.2. *La complejidad de la zona de* γ_0 *en un arreglo de n curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo s puntos es*

$O(\lambda_{s+2}(n))$, suponiendo que γ_0 intersecta cada curva del arreglo en a lo sumo una cantidad constante de puntos ([13], pág. 125).

De este teorema se deduce una cota superior para la construcción del arreglo $\mathcal{A}(\Gamma)$ usando, por ejemplo, un algoritmo incremental:

Teorema 3.1.3. *El arreglo de un conjunto de n curvas de Jordan no acotadas en el plano que se intersectan dos a dos en a lo sumo s puntos puede ser calculado en tiempo $O(n\,\lambda_{s+2}(n))$ ([13], pág. 172).*

Consideremos ahora el caso particular en que $\Gamma \cup \{\gamma_0\}$ es un conjunto de curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo dos puntos. Gracias a los teoremas conque se cuenta, la cota para la complejidad de la zona de γ_0 en $\mathcal{A}(\Gamma)$ es en este caso $O(\lambda_4(n))$. Consecuencia de esto es que $\mathcal{A}(\Gamma)$ puede construirse en tiempo $O(n\,\lambda_4(n))$.

Probaremos que es posible construir $\mathcal{A}(\Gamma)$ en tiempo $O(n\,\lambda_3(n))$. Sabiendo que $\lambda_4(n) = \Theta(n\,2^{\alpha(n)})$ ([13], pág. 56) y que $\lambda_3(n) = \Theta(n\,\alpha(n))$ ([13], pág. 29), habremos mejorado la complejidad del tiempo de construcción de estos arreglos.

3.2 Notación y Definiciones

Sea entonces Γ un conjunto de n curvas de Jordan no acotadas en el plano que se intersectan dos a dos en a lo sumo dos puntos. Llamaremos $\mathcal{A}(\Gamma)$ al arreglo de estas curvas. Sea γ_0 una curva de Jordan no acotada en el plano que se intersecta con cada curva de Γ en a lo sumo dos puntos. Asumimos que $\Gamma \cup \{\gamma_0\}$ está en posición general, es decir, todas las intersecciones son transversales y no hay tres curvas que se corten en un mismo punto.

La zona de γ_0 en $\mathcal{A}(\Gamma)$, denotada por $z(\gamma_0)$, es el conjunto de caras de $\mathcal{A}(\Gamma)$ intersectadas por γ_0. La complejidad de $z(\gamma_0)$ viene dada por el número total de aristas en las fronteras de las caras de $z(\gamma_0)$.

Cualquier curva de Γ que tenga intersección con γ_0 puede ser descompuesta por γ_0 en dos o tres tramos conexos. Llamaremos *ramas* a los tramos no acotados. Las ramas se considerarán orientadas, siendo el punto de partida de una rama el punto de intersección con γ_0. A este punto lo llamaremos *base de la rama*. La base de una rama r será denotada por $\overline{r}$.

Cada curva de Γ que intersecte a γ_0 en dos puntos dará lugar a dos ramas. Cuando las ramas de todas las curvas de Γ que se intersectan con γ_0 en dos puntos estén hacia un mismo lado de γ_0 diremos que γ_0 es una curva *pseudo-convexa* con respecto a Γ y llamaremos *zona exterior* a la parte de $z(\gamma_0)$ que queda hacia ese lado de γ_0. En la figura 3.2 tenemos una ilustración. La región sombreada es la zona exterior, a la que denotaremos $z^+(\gamma_0)$.

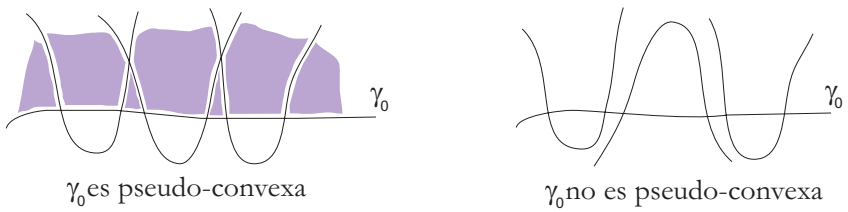


Figura 3.2: Pseudo-convexidad.

La complejidad de la zona exterior se define como el número total de aristas pertenecientes a la misma. Se considera que una arista pertenece a la zona exterior si pertenece a alguna de las caras de $z^+(\gamma_0)$ pero no a γ_0.

Orientemos γ_0 de manera que la zona exterior quede a su izquierda. Una vez fijada la orientación, podemos establecer una relación de orden total entre las ramas y las caras de la zona exterior. Sean r, s, $\mathcal{F}$ y $\mathcal{G}$ respectivamente dos ramas y dos caras de $z^+(\gamma_0)$. Decimos que

- $\mathcal{F}$ está a la izquierda de $\mathcal{G}$ y lo denotamos por $\mathcal{F} < \mathcal{G}$ si $\mathcal{F} \neq \mathcal{G}$ y $\mathcal{F}$ aparece en γ_0 antes que $\mathcal{G}$ al recorrer γ_0 según su orientación.

- r está a la izquierda de $\mathcal{F}$ ($r < \mathcal{F}$) si $\overline{r}$ no aparece en γ_0 después que $\mathcal{F}$.
- r está a la izquierda de s ($r < s$) si $r \neq s$ y $\overline{r}$ aparece en γ_0 antes que $\overline{s}$.

Utilizaremos la notación $|r \cap s|$ para referirnos al número de intersecciones entre r y s.

Como ya sabemos, la frontera de una cara está formada por todas las aristas que la conforman. Dada una cara de la zona exterior, vamos a llamar *frontera exterior* a cada una de las componentes conexas obtenidas al eliminar de la frontera de la cara las aristas que pertenecen a γ_0. Consecuencia de esta definición es que los extremos de las fronteras exteriores pertenecen a γ_0.

De ahora en adelante supondremos que γ_0 es pseudo-convexa con respecto a Γ y trabajaremos con la zona exterior de γ_0. También consideraremos, sin pérdida de generalidad, que todas las caras de la zona exterior, a excepción de la primera y la última, son acotadas. En caso de que este supuesto no se cumpla, bastará con añadir al conjunto Γ una nueva curva que esté a la izquierda de γ_0 y que intersecte únicamente a las caras no acotadas de $\mathcal{A}(\Gamma)$.

3.3 Preliminares

El algoritmo que construye $\mathcal{A}(\Gamma)$ en tiempo $O(n\,\lambda_3(n))$ está basado en tres lemas. El primer lema nos dice que los extremos de cada frontera exterior definen una arista de γ_0 en $\mathcal{A}(\Gamma \cup \{\gamma_0\})$. Esto garantiza que haciendo el recorrido de las fronteras exteriores, según van apareciendo las correspondientes caras en γ_0, obtenemos el orden en que γ_0 se intersecta con las curvas de Γ.

De las definiciones de frontera exterior y complejidad de la zona exterior se deduce que el recorrido de las fronteras exteriores se hace en tiempo proporcional a la complejidad de la zona exterior. El segundo lema demuestra que la complejidad de $z^+(\gamma_0)$ en $\mathcal{A}(\Gamma)$ es $O(\lambda_3(n))$.

Por último, el tercer lema nos asegura que podemos hallar siempre un orden de inserción de las curvas en el arreglo tal que la i-ésima curva que se inserta es siempre pseudo-convexa con respecto a las $i-1$ ya insertadas.

Ya al final del capítulo y a modo de generalización vemos que si las curvas del conjunto Γ se intersectan dos a dos en a lo sumo s puntos, γ_0 intersecta a cada una de ellas en a lo sumo dos puntos y γ_0 es pseudo-convexa con respecto a Γ entonces γ_0 puede insertarse en $\mathcal{A}(\Gamma)$ en tiempo $O(\lambda_{s+1}(n))$. Con los resultados que se tenían hasta ahora ([13], pág. 125) la inserción de γ_0 hubiese costado en este caso $O(\lambda_{s+2}(n))$.

3.4 La Frontera Exterior

Lema 3.4.1. *Los extremos de una frontera exterior definen una arista de γ_0 en el arreglo $\mathcal{A}(\Gamma \cup \{\gamma_0\})$.*

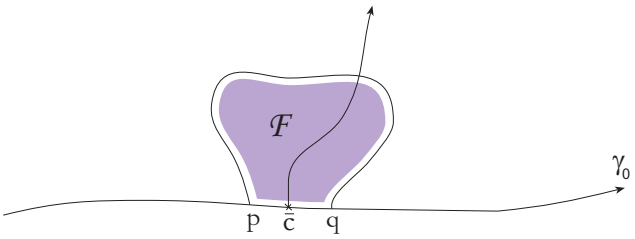


Figura 3.3: Ilustración.

Demostración. Sea $\mathcal{F}^*$ la frontera exterior de una cara de la zona exterior. Sean p y q los extremos de $\mathcal{F}^*$. Supongamos, por reducción al absurdo, que $\overline{pq}$ no es una arista de γ_0 en el arreglo antes mencionado. En este caso, $\overline{pq}$ va a contener en su interior al menos un punto de intersección, llamémosle c. Como $\overline{pq}$ es un subarco de γ_0, c tiene que ser la base de alguna rama. Como las ramas son no acotadas, c y $\mathcal{F}^*$ tienen que intersectarse entre sí. Esto contradice el hecho de que $\mathcal{F}^*$ sea frontera exterior de una cara. Véase la figura 3.3.

□

Este lema nos va a permitir insertar γ_0 en $\mathcal{A}(\Gamma)$ en tiempo proporcional a la complejidad de la zona exterior de γ_0: recorriendo las sucesivas fronteras exteriores obtenemos los extremos de las aristas que componen γ_0 una vez que está en el arreglo.

3.5 Complejidad de la Zona Exterior

En esta sección demostramos que la zona exterior de γ_0 en $\mathcal{A}(\Gamma)$ tiene complejidad $O(\lambda_3(n))$. Por definición, acotar la complejidad de la zona exterior equivale a acotar el número de aristas de la misma.

Para facilitar la acotación del número de aristas de $z^+(\gamma_0)$ las dividimos en grupos. Tendremos así tres grupos: las neutrales, las izquierdas y las derechas. Las llamaremos también aristas de tipo N, de tipo I y de tipo D respectivamente. Sea e una arista de $z^+(\gamma_0)$ tal que su curva soporte se intersecta con γ_0. Sea r la rama soporte de e y $\mathcal{F}$ una de las dos posibles caras en las que e aparece.

Definición 3.5.1. *Decimos que la arista e es izquierda si $r < \mathcal{F}$ y es derecha si $\mathcal{F} < r$.*

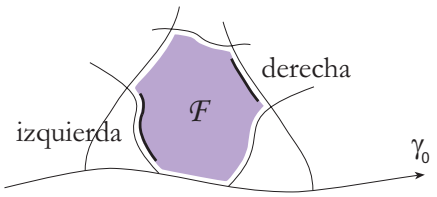


Figura 3.4: Aristas izquierdas y derechas de $\mathcal{F}$.

Observación 3.5.1. De la definición anterior se deduce que una rama sólo puede aportar aristas I a aquellas caras que tiene a su derecha. De manera simétrica, una rama sólo puede aportar aristas D a las caras que tiene a su izquierda.

Definición 3.5.2. *Decimos que e es neutral si la curva a la que pertenece no tiene intersección con* γ_0.

Vista la clasificación de las aristas vamos ahora a contarlas. La idea general es visitar las aristas de la zona exterior de manera ordenada y anotar, por cada arista visitada, un símbolo. Una vez que tengamos la lista comprobamos que se trata de una secuencia de Davenport-Schinzel.

Para visitar todas las aristas nos movemos de cara en cara recorriendo la frontera exterior de las mismas en sentido horario. El orden de visita de las caras será el orden en el que aparecen al recorrer γ_0 según su orientación. Recordemos ahora los requisitos que una lista de símbolos tiene que cumplir para ser una secuencia de Davenport-Schinzel.

Sea $U = \langle u_1, u_2, \ldots, u_m \rangle$ la secuencia de símbolos obtenida. Para ser una secuencia (n, s) de Davenport-Schinzel, U tiene que tener las siguientes propiedades (ver [13], pág. 1):

(i) contener a lo sumo n símbolos distintos.
(ii) $u_i \neq u_{i+1}$ para todo $1 \leq i < m$.
(iii) no pueden existir $s+2$ índices $1 \leq i_1 < i_2 < \ldots < i_{s+2} \leq m$ tales que

$$u_{i_1} = u_{i_3} = u_{i_5} \ldots = a,$$
$$u_{i_2} = u_{i_4} = u_{i_6} \ldots = b \text{ y}$$
$$a \neq b.$$

El símbolo que se anotará por cada arista de la zona exterior va a ser el que identifique a su rama o curva soporte. Como en la zona exterior una misma rama puede ser soporte de aristas I y D, será necesario utilizar símbolos diferentes para cada lado de la rama: un símbolo para el lado soporte de aristas I y otro para el lado soporte de aristas D. De este modo, n curvas dan lugar a $4n$ símbolos a lo sumo (dos por cada una de las 2n posibles ramas). Antes de probar que U es una secuencia de Davenport-Schinzel vamos a ver la siguiente propiedad.

Propiedad 3.5.1. *Las aristas izquierdas aparecen en cualquier cara de la zona exterior antes que las derechas.*

Demostración. Sea $\mathcal{F}^*$ la frontera exterior de una cara cualquiera de la zona exterior. Sean e y f dos aristas de $\mathcal{F}^*$ de tipo izquierda y derecha respectivamente. Supongamos, razonando por reducción al absurdo, que f aparece antes que e en la cara. Llamemos γ_e y γ_f a las ramas soporte de e y f respectivamente. Si recorremos las ramas desde sus bases tenemos que, para que f aparezca antes que e, γ_e y γ_f tienen que intersectarse antes de llegar a $\mathcal{F}^*$ ya que $\gamma_e < \gamma_f$. Ahora, una vez que han salido de $\mathcal{F}^*$, cada rama va a parar a una región que está limitada por γ_0, $\mathcal{F}^*$ y la otra rama. En la figura 3.5 tenemos un ejemplo en el que aparecen sombreadas la cara a la que e y f pertenecen y la región a la que llega γ_f después de pasar por f.

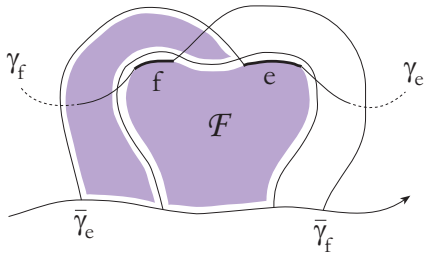


Figura 3.5: Ilustración.

Como las ramas son no acotadas y pertenecen a curvas de Jordan, no pueden quedarse dentro de una región acotada, luego, tienen que salir. Como las ramas no pueden intersectarse con γ_0 en un punto que no sea la base de la rama y como tampoco pueden entrar en una cara (en este caso $\mathcal{F}^*$ dejaría de ser la frontera exterior de esa cara) para salir tendrían que intersectar a la otra rama, dando lugar a dos nuevos puntos de intersección que, sumados al anterior, harían un total de tres. Esto, sin embargo, contradice el hecho de que dos ramas se intersectan en a lo sumo dos puntos.

□

Propiedad 3.5.2. *U es una secuencia $(4n, 3)$ de Davenport-Schinzel.*

Demostración. Veremos que U cumple con cada uno de los requisitos.

- requisito (i): los $4n$ símbolos distintos vienen de los nombres dados a las a lo sumo $2n$ ramas.

- requisito (ii): dentro de una misma cara no podemos encontrar dos aristas consecutivas aportadas por una misma rama (o curva, para el caso de las aristas neutrales). Esto se debe a que las intersecciones se dan de manera transversal. Al pasar de una cara a otra, tampoco se puede repetir el mismo símbolo, ya que se pasa de un lado de la rama al otro, y hemos utilizado símbolos diferentes para cada lado. U cumple entonces con el segundo requisito.

- requisito (iii): este requisito nos dice que U no puede tener subcadenas de longitud 5 en las que dos símbolos alternen de manera sucesiva. En otras palabras, no existen γ_i y γ_j que den lugar a subcadenas de U de la forma $\gamma_i, \gamma_j, \gamma_i, \gamma_j, \gamma_i$. Para ver que U cumple también con este requisito consideraremos los siguientes casos:

  Caso (1): γ_i y γ_j son símbolos correspondientes a aristas neutrales.

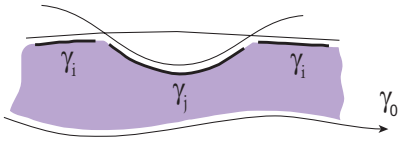


Figura 3.6: Alternancia entre curvas soporte de aristas neutrales.

  Como estas curvas no tienen intersección con γ_0, para que alternen en U tienen que intersectarse; ver figura 3.6. Cada punto de intersección da lugar a una alternancia entre γ_i y γ_j. Como el número máximo de intersecciones entre dos curvas es 2 tendremos a lo sumo 2 alternancias, que dan lugar a subcadenas γ_i,γ_j,γ_i de longitud máxima 3.

Caso (2): γ_i y γ_j son símbolos correspondientes a aristas derechas de dos ramas. Supongamos que $\gamma_i < \gamma_j$. Sean $\mathcal{F}^{'}$, $\mathcal{F}^{''}$ y $\mathcal{F}^{'''}$ el conjunto de caras de $z^{+}(\gamma_0)$ tales que: $\mathcal{F}^{'} < \gamma_i < \mathcal{F}^{''} < \gamma_j < \mathcal{F}^{'''}$.

Gracias a la observación 3.5.1 sabemos que γ_i y γ_j sólo pueden alternar en las caras de $\mathcal{F}^{'}$, que aparecen sombreadas en el ejemplo de la figura 3.19.

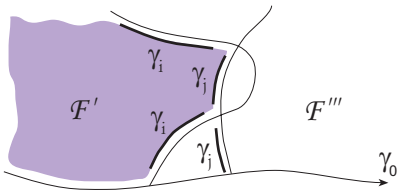


Figura 3.7: Alternancia entre lados soporte de aristas derechas.

Como γ_i y γ_j etiquetan aristas del mismo tipo, para que estos símbolos alternen en $\mathcal{F}^{'}$, las ramas a las que están asociados tienen que intersectarse entre sí. Las subcadenas donde γ_i y γ_j alternan tendrán entonces longitud máxima 3 y terminarán en γ_i por cumplirse que $\gamma_i < \gamma_j$ y por tratarse de aristas derechas. Consecuencia también de la observación 3.5.1 es el hecho de que en las caras de $\mathcal{F}^{''}$ no tendremos ninguna aparición del símbolo γ_i pero sí tendremos al menos una de γ_j. Como los conjuntos $\mathcal{F}^{'}$ y $\mathcal{F}^{''}$ aparecen contiguos en γ_0, se tiene entonces una nueva alternancia, que será también la última, dado que en $\mathcal{F}^{'''}$ no encontraremos aristas con etiqueta γ_i ni γ_j. Finalmente, las subcadenas de U donde estos símbolos alternan tienen en este caso longitud máxima 4 y son de la forma γ_i,γ_j,γ_i,γ_j.

Caso (3): γ_i y γ_j son símbolos correspondientes a aristas izquierdas. Este caso es simétrico con respecto al anterior.

Caso (4): γ_i es el símbolo correspondiente a una arista neutral y γ_j es el símbolo correspondiente a una arista derecha.

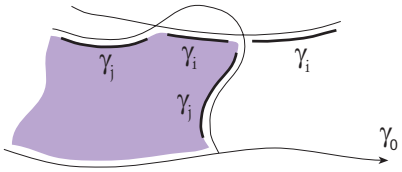


Figura 3.8: Las caras donde γ_i y γ_j pueden alternar.

Sabemos que γ_i y γ_j sólo pueden alternar en las caras que están a la izquierda de γ_j, que son las que aparecen sombreadas en la figura 3.20. Teniendo en cuenta los argumentos utilizados en los otros casos llegamos a subcadenas de longitud máxima 4 que serán de la forma γ_j,γ_i,γ_j,γ_i.

Caso (5): γ_i es el símbolo correspondiente a una arista neutral y γ_j es el símbolo correspondiente a una arista izquierda. Este caso es simétrico con respecto al anterior.

Caso (6): γ_i es el símbolo correspondiente a una arista izquierda y γ_j es el símbolo correspondiente a una arista derecha. En este apartado la longitud de las subcadenas varía dependiendo del orden en que las ramas aparecen en γ_0. Tenemos entonces dos opciones.

Opción (a): $\gamma_j < \gamma_i$.

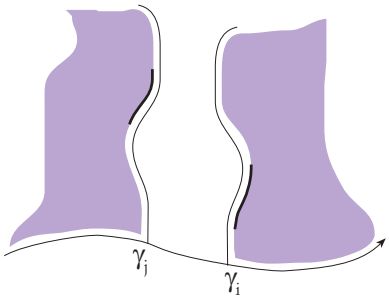


Figura 3.9: No existen caras de la zona exterior donde γ_i y γ_j puedan alternar.

Como vemos en la figura 3.9, en este caso no existe una cara de la zona exterior donde γ_i y γ_j alternen ya que ninguna cara de la zona exterior puede aparecer a la vez antes de γ_j y después de γ_i (las caras que aparecen antes de γ_j son las únicas que pueden contener aristas γ_j y las que aparecen después de γ_i son las únicas que pueden contener aristas γ_i). La única alternancia que existe se da entonces al pasar de una cara a otra. Las subcadenas alternas tienen en este caso longitud máxima 2 y son de la forma γ_j,γ_i.

Opción (b): $\gamma_i < \gamma_j$.

Si llamamos $\mathcal{F}'$, $\mathcal{F}''$ y $\mathcal{F}'''$ al conjunto de caras tales que: $\mathcal{F}' < \gamma_i < \mathcal{F}'' < \gamma_j < \mathcal{F}'''$ se tiene que γ_i y γ_j sólo pueden alternar en las caras de $\mathcal{F}''$. Véase que $\mathcal{F}'$ no contiene aristas γ_i ni $\mathcal{F}'''$ aristas γ_j. La primera alternancia entre γ_i y γ_j en U se produce entonces al pasar de $\mathcal{F}'$ a $\mathcal{F}''$ y será de la forma γ_j,γ_i, mientras que la última se da al pasar de $\mathcal{F}''$ a $\mathcal{F}'''$, con la subcadena γ_j,γ_i.

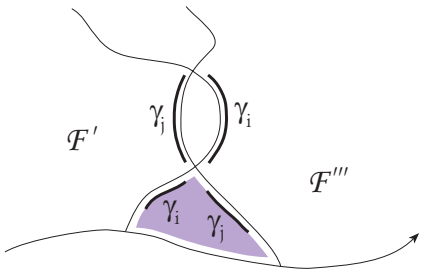


Figura 3.10: Las caras donde γ_i y γ_j alternan aparecen sombredas en la figura.

En $\mathcal{F}''$ tenemos que, según la propiedad 3.5.1, las aristas de etiqueta γ_i aparecen todas antes que las de etiqueta γ_j. Esto quiere decir que el número de alternancias en $\mathcal{F}''$ será solamente uno y la subcadena será del tipo γ_i,γ_j. Como en $\mathcal{F}'$ tenemos solamente aristas γ_j y en el conjunto $\mathcal{F}'''$, γ_i, el número total de alternancias será a lo sumo 3, con subcadenas de longitud máxima 4 que son de la forma γ_j,γ_i,γ_j,γ_i.

Llegamos así a lo que queríamos demostrar. No existen subcadenas en U de longitud 5 en las que dos símbolos alternen. U es, por lo tanto, una secuencia $(4n, 3)$ de Davenport-Schinzel, siendo $\lambda_3(4n)$ la longitud máxima de U ([13], pág. 1).

□

Lema 3.5.1. *La zona exterior de γ_0 tiene complejidad $O(\lambda_3(n))$.*

Demostración. Dada la propiedad 3.5.2, basta con ver que $\lambda_3(4n) = O(\lambda_3(n))$.
□

3.6 El Orden de Inserción de las Curvas

Con este tercer y último lema vamos a ver que las curvas de Γ siempre pueden ser ordenadas de manera tal que al añadirlas al arreglo de una en una en el orden dado la i-ésima curva es pseudo-convexa con respecto a las $i-1$ primeras, para $2 \leq i \leq n$.

Recordando la definición de pseudo-convexidad esto quiere decir que las curvas de Γ pueden ser ordenadas de modo que para $2 \leq i \leq n$ se cumpla que en uno de los lados de la i-ésima curva encontremos tramos no acotados de cada una de las $i-1$ anteriores. Dicho en otras palabras, existe al menos un orden para las curvas en el que cada una tiene su zona exterior en el momento de ser insertada. Veremos primero el procedimiento para obtener la lista ordenada de curvas y luego que este procedimiento garantiza la pseudo-convexidad de cada curva.

Para establecer el orden se ha tomado un acuerdo: la ordenación se llevará a cabo de modo que la zona exterior de una curva quede siempre a su izquierda. Esto quiere decir que al final cada curva tendrá asignada una orientación. En general, la zona exterior podría situarse en cualquier lado pero en ese caso las curvas tendrían que indicar, además de su orientación, en cuál de los dos lados quedó su zona exterior.

De manera general, la idea para ordenar es la siguiente. Se fija una curva γ cualquiera y las demás se sitúan con respecto a ella, dando lugar así a dos conjuntos que es preciso ordenar: el conjunto de las curvas que se insertarán antes que γ en el arreglo y el conjunto de las que se insertarán después. Como es de esperar, las curvas que se insertan antes de γ no son otras que aquellas que aportan tramos no acotados al lado izquiedo de γ, lo cual significa que, antes de empezar a situar curvas con respecto a γ, ésta tiene que estar ya orientada. Los pasos de este procedimiento recursivo se detallan a continuación.

Procedimiento para Ordenar el Conjunto Γ;

(1) Si $\Gamma = \emptyset$ no hacer nada. Devolver $\emptyset$ como lista ordenada.

(2) Seleccionamos al azar una curva $\gamma \in \Gamma$.

(3) Orientamos γ en un sentido cualquiera, si no ha sido ya orientada.

(4) Dividimos el resto de las curvas de Γ - $\{\gamma\}$ en dos conjuntos $\mathcal{C}_1(\gamma)$ y $\mathcal{C}_2(\gamma)$ que contendrán las curvas de Γ - $\{\gamma\}$ a insertar antes y después de γ respectivamente. Así, para cada curva $\delta \in \Gamma - \{\gamma\}$ hacemos lo siguiente:

- Incluimos a δ en $\mathcal{C}_1(\gamma)$ si se da una de las tres situaciones siguientes,

  $|\delta \cap \gamma| = 0$ y δ se encuentra a la izquierda de γ.

  $|\delta \cap \gamma| = 1$.

  $|\delta \cap \gamma| = 2$ y las ramas de δ quedan a la izquierda de γ.

- Incluimos a δ en $\mathcal{C}_2(\gamma)$ si no fue incluida en $\mathcal{C}_1(\gamma)$, o sea, si

  $|\delta \cap \gamma| = 0$ y δ se encuentra a la derecha de γ o si

  $|\delta \cap \gamma| = 2$ y las ramas de δ quedan a la derecha de γ.

(5) Orientamos las curvas de $\mathcal{C}_2(\gamma)$ en función de la orientación dada a γ de modo que se cumpla el acuerdo tomado: la zona exterior de una curva está a su izquierda. En otras palabras, la orientación de estas curvas tiene que ser tal que el lado donde queden los tramos no acotados de γ sea el lado izquierdo de cada una de ellas.

(6) Repetimos el procedimiento para $\mathcal{C}_1(\gamma)$ y $\mathcal{C}_2(\gamma)$, obteniendo las listas ordenadas $\mathcal{L}_1(\gamma)$ y $\mathcal{L}_2(\gamma)$ respectivamente.

(7) El orden de las curvas en Γ es $\mathcal{L}_1(\gamma) + [\gamma] + \mathcal{L}_2(\gamma)$, donde "+" representa la concatenación de listas.

Definición 3.6.1. *A la curva γ que sirve como curva de referencia dentro de un conjunto para situar las demás curvas se le llamará de este modo,* curva de referencia.

Definición 3.6.2. *A los conjuntos $\mathcal{C}_1(\gamma)$ y $\mathcal{C}_2(\gamma)$ a que γ da lugar se les llamará* conjuntos asociados.

Observación 3.6.1. De los pasos del procedimiento se deduce que toda curva es en algún momento curva de referencia y tiene asociados, por lo tanto, dos conjuntos (un conjunto asociado es igual, en última instancia, al conjunto vacío).

Véase que el procedimiento orienta las curvas de $\mathcal{C}_2(\gamma)$, paso (5), pero no las de $\mathcal{C}_1(\gamma)$. Como ya se menciona en el paso (5), esto está relacionado con el hecho de que se quiere tener siempre la zona exterior en el lado izquierdo de las curvas. Como las curvas de $\mathcal{C}_2(\gamma)$ van a ser insertadas en el arreglo después de γ, al llegar el turno de cada una de ellas, γ está ya en el arreglo, por lo que estas curvas tienen que "saber" a qué lado está su zona exterior. En la figura 3.11 tenemos un ejemplo.

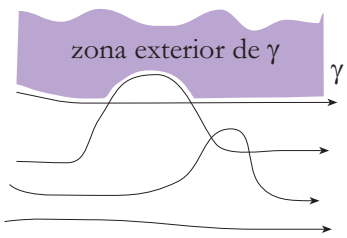


Figura 3.11: γ determina la orientación de las curvas en $\mathcal{C}_2(\gamma)$.

Con las curvas de $\mathcal{C}_1(\gamma)$ no se tiene esta preocupación ya que, sea cual sea la orientación que se le dé a γ, al ser insertadas en el arreglo, γ aún no ha aparecido. La única herencia que estas curvas reciben de γ es que la orientación de γ las obliga a ser insertadas antes en el arreglo.

Para facilitar la demostración de lo que queremos, (la pseudo-convexidad de cada curva con respecto a las que le preceden dentro del orden) vamos a introducir nuevos conceptos y a probar algunas propiedades que cumple la lista ordenada. Para empezar, consideraremos el infinito representado por una pseudo-circunferencia. La pseudo-circunferencia será entonces lo suficientemente grande como para contener todos los puntos de intersección entre las curvas e intersectar cada curva en sólo dos puntos.

Definición 3.6.3. *A los dos puntos de intersección entre una curva $\gamma \in \Gamma$ y la pseudo-circunferencia los llamaremos* extremos *de γ. Estos puntos se considerarán puntos en el infinito.*

Definición 3.6.4. *Cada una de las dos partes en que una curva divide a la pseudo-circunferencia se llamará* bóveda. *Una vez orientada γ, hablaremos de la bóveda izquierda y de la bóveda derecha de γ.*

Volvamos a la orientación que γ determina en las curvas de $\mathcal{C}_2(\gamma)$. Una vez vistas las definiciones anteriores tenemos que, por construcción, los dos extremos de todas las curvas de $\mathcal{C}_2(\gamma)$ están en la bóveda derecha de γ. La orientación dada por γ a estas curvas tiene que ser tal que los dos extremos de γ están en la bóveda izquierda de cada una. Consecuencia de esto es que, una vez fijada la orientación de las curvas en $\mathcal{C}_2(\gamma)$, la bóveda izquierda de γ está contenida en la bóveda izquierda de cada una de ellas. En la figura 3.12 tenemos una ilustración.

Figura 3.12: Bóveda izquierda de γ contenida en bóveda izquierda de δ, para $\delta \in \mathcal{C}_2(\gamma)$.

Antes de seguir probaremos que el procedimiento de ordenación es consistente: la orientación que una curva de referencia γ determina en su conjunto asociado $\mathcal{C}_2(\gamma)$ no impide la posterior ordenación de este conjunto. Para demostrar esta propiedad vamos a tener en cuenta que si ρ es una curva de $\mathcal{C}_2(\gamma)$ entonces su conjunto asociado $\mathcal{C}_2(\rho)$ está contenido dentro del conjunto asociado de γ $\mathcal{C}_2(\gamma)$. Utilizaremos la notación $\widehat{\gamma_i}$ para referirnos a la bóveda izquierda de γ_i y la notación $\widehat{\gamma_i} \subset \widehat{\gamma_j}$ para expresar que la bóveda izquierda de γ_i está contenida en la bóveda izquierda de γ_j.

Propiedad 3.6.1. *Si γ es una curva cualquiera y ρ una curva que pertenece a $\mathcal{C}_2(\gamma)$ entonces, la orientación fijada por γ en las curvas de $\mathcal{C}_2(\rho)$ es la misma que la inducida por ρ en este conjunto.*

Demostración. Sea $\psi \in \mathcal{C}_2(\rho)$. La posición relativa de las tres curvas dentro de la lista ordenada final es entonces $\ldots\ \gamma\ \ldots\ \rho\ \ldots\ \psi\ \ldots$ Como la orientación de γ determina la orientación en las curvas de su conjunto asociado $\mathcal{C}_2(\gamma)$ y como ρ y ψ pertenecen a este conjunto, una vez fijada la orientación de γ y por consiguiente, la de ρ y ψ se cumplirá que,

$$\text{(i) } \widehat{\gamma} \subset \widehat{\rho} \qquad \text{y}$$
$$\text{(ii) } \widehat{\gamma} \subset \widehat{\psi}.$$

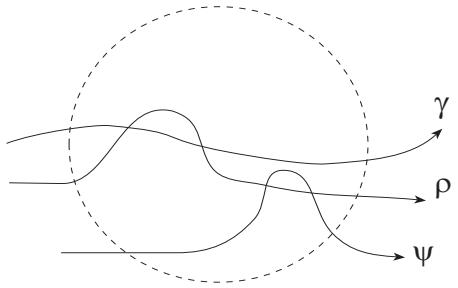


Figura 3.13: La bóveda izquierda de γ está contenida en las bóvedas izquierdas de ρ y ψ.

En la figura 3.13 tenemos un ejemplo. Supongamos ahora, por reducción al absurdo, que la orientación que ρ induce en ψ, por cumplirse $\psi \in \mathcal{C}_2(\rho)$, es diferente a la que γ fijó. La orientación que ρ induce en ψ tiene que ser tal que $\widehat{\rho} \subset \widehat{\psi}$. Si esta orientación es diferente a la dada por γ es porque con esta orientación, $\widehat{\gamma}$ queda en la bóveda derecha de ψ. Llegamos entonces a que $\widehat{\gamma}$ y $\widehat{\rho}$ están cada una a un lado diferente de ψ, es decir, $\widehat{\rho} \subset \widehat{\psi} \subset \widehat{\gamma}$, lo que entra en contradicción con (i).

□

La propiedad que vamos a ver a continuación sirve de base al lema que queremos demostrar en esta sección. Las demostraciones que siguen se hacen más fáciles si redefinimos la pseudo-convexidad de una curva teniendo en cuenta el concepto de bóveda.

Definición 3.6.5. *Decimos que γ es pseudo-convexa con respecto a un determinado conjunto de curvas si cada una de las mismas tiene al menos un punto de intersección con $\widehat{\gamma}$.*

Propiedad 3.6.2. *Dados una curva γ y sus dos conjuntos asociados se tiene que:*

(i) γ es pseudo-convexa con respecto a las curvas de $\mathcal{C}_1(\gamma)$.
(ii) Las curvas de $\mathcal{C}_2(\gamma)$ son pseudo-convexas con respecto a γ.
(iii) Las curvas de $\mathcal{C}_2(\gamma)$ son pseudo-convexas con respecto a las de $\mathcal{C}_1(\gamma)$.

Demostración. Los puntos (i) y (ii) se cumplen por construcción. Para ver el punto (iii), sea $\gamma_1 \in \mathcal{C}_1(\gamma)$. Sabemos entonces que, por construcción, γ_1 tiene al menos un punto de intersección con $\widehat{\gamma}$. Sea $\gamma_2 \in \mathcal{C}_2(\gamma)$. Por construcción, $\widehat{\gamma} \subset \widehat{\gamma_2}$. Esto quiere decir que γ_1 tiene al menos un punto de intersección con $\widehat{\gamma_2}$. Luego, por la definición 3.6.5, γ_2 es pseudo-convexa con respecto a γ_1. Llegamos así a lo que queríamos demostrar.

□

Teniendo en cuenta todo lo visto en esta sección podemos probar ya el tercer y último lema en el que está basado el algoritmo.

Lema 3.6.1. *Si γ_1, γ_2, ..., γ_n es el orden hallado con el procedimiento descrito anteriormente entonces γ_i es una curva pseudo-convexa con respecto a γ_1, γ_2, ..., γ_{i-1} para todo $2 \leq i \leq n$.*

Demostración. La demostración consiste en tomar una curva cualquiera del conjunto γ_1, γ_2, ..., γ_{i-1} y ver que γ_i es pseudo-convexa con respecto a la misma. Sea entonces γ_j una curva cualquiera tal que $j < i$. Sabemos que, por construcción, existe una curva de referencia γ_k, $j \leq k \leq i$, que separa a γ_i de γ_j, dándose una de las tres situaciones siguientes:

(i) $\gamma_k = \gamma_i$ y $\gamma_j \in \mathcal{C}_1(\gamma_k)$.
(ii) $\gamma_k = \gamma_j$ y $\gamma_i \in \mathcal{C}_2(\gamma_k)$.
(iii) $\gamma_j \in \mathcal{C}_1(\gamma_k)$ y $\gamma_i \in \mathcal{C}_2(\gamma_k)$.

Teniendo en cuenta la propiedad 3.6.2 llegamos a que, en cualquiera de los tres casos, γ_i es pseudo-convexa con respecto a γ_j. Dado que esto se cumple para todo $j < i$, tenemos lo que queríamos demostrar. □

3.7 Resultados

La propiedad de pseudo-convexidad vista en el lema 3.6.1 nos garantiza que cada curva va a tener su zona exterior. El lema 3.5.1 nos da una cota superior para la

complejidad de esta zona exterior y el primer lema, el 3.4.1, nos dice que podemos insertar cada curva en tiempo proporcional a la complejidad de su zona exterior. Todo esto trae como consecuencia que el tiempo de construcción del arreglo de las n curvas de Γ sea a lo sumo n veces la complejidad de la zona exterior.

Teorema 3.7.1. *El arreglo $\mathcal{A}(\Gamma)$ de curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo dos puntos puede ser construido en tiempo $O(n\,\lambda_3(n))$.*

3.8 Construcción del Arreglo

A grandes rasgos, los pasos para construir el arreglo son sólo dos, como vemos en el recuadro siguiente.

Algoritmo para construir A(Γ)

[1] Ordenar el conjunto Γ utilizando el procedimiento visto en la sección 1.6.

[2] Construir $\mathcal{A}(\Gamma)$ utilizando un algoritmo incremental donde en lugar de trabajar con la zona de las curvas se trabaja con la que hemos llamado zona exterior.

El segundo paso es el que determina la complejidad total del algoritmo: $O(n\,\lambda_3(n)) = O(n^2\,\alpha(n))$. Como en el método incremental las curvas se insertan en el arreglo de una en una, la inserción de cada curva depende de la complejidad de su zona exterior: $O(\lambda_3(n))$.

El primer paso tiene complejidad $O(n^2)$ en el peor caso. Véase que el procedimiento que se utiliza para la ordenación es similar al *Quicksort* [8]. Como se sabe, el peor caso se da cuando las sucesivas divisiones quedan desequilibradas,

o sea, la división de cada conjunto da lugar a dos subconjuntos donde uno tiene tamaño $O(n)$ y el otro $O(1)$.

3.9 Cota Inferior

Sabemos que dado un conjunto de n segmentos, la envolvente inferior del mismo tiene complejidad $\Omega(\lambda_3(n))$ ([13], pág. 112). Este resultado podemos parafrasearlo del siguiente modo:

Teorema 3.9.1. *Para cualquier entero n podemos obtener un conjunto de n segmentos de recta en el plano cuya envolvente inferior conste de $\Omega(n\,\alpha(n))$ subsegmentos.*

Veremos que la complejidad de la zona $z^+(\gamma_0)$ tiene también cota inferior $\lambda_3(n) = \Theta(n\,\alpha(n))$. La demostración consiste en ver que dado un conjunto $\mathcal{Q}$ de segmentos éste puede transformarse siempre en un conjunto $\Gamma(\mathcal{Q})$ de curvas que se intersectan dos a dos en a lo sumo dos puntos de manera tal que al considerar el arreglo $\mathcal{A}(\Gamma(\mathcal{Q}))$, la envolvente inferior de $\mathcal{Q}$ forme parte de la zona exterior de una curva γ_0 elegida adecuadamente ($\gamma_0 \notin \Gamma(\mathcal{Q})$). Sea entonces $\mathcal{Q}$ un conjunto de n segmentos cuya envolvente inferior tiene complejidad $\Omega(\lambda_3(n))$ y γ_0 una recta horizontal situada por debajo de $\mathcal{Q}$.

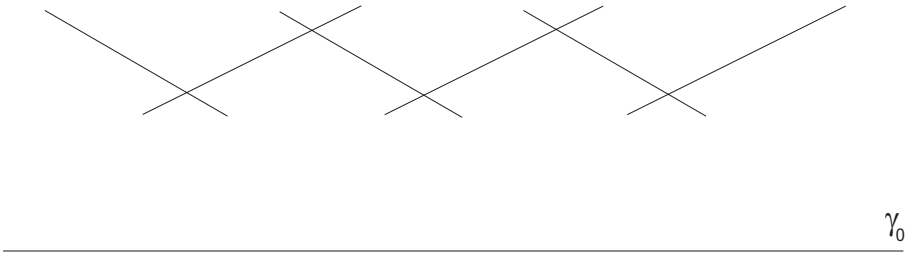


Figura 3.14: El conjunto $\mathcal{Q}$ y γ_0.

Nota 3.9.1. El conjunto $\mathcal{Q}$ lo seleccionamos de manera tal que no haya tres segmentos pasando por un mismo punto, que no haya dos extremos de segmento con igual abscisa ni un extremo de segmento perteneciendo al interior de otro.

Para transformar $\mathcal{Q}$ en el conjunto de curvas $\Gamma(\mathcal{Q})$ vamos a añadir rayos verticales en los extremos de los segmentos de manera que cada segmento dé lugar a una curva. Los rayos verticales se van a añadir de manera tal que las curvas obtenidas se intersecten dos a dos en a lo sumo dos puntos y que cada curva se intersecte con γ_0 en un único punto. Utilizaremos la notación $\mathcal{Q}^*$ para referirnos a la envolvente inferior de $\mathcal{Q}$. Sea entonces s un segmento cualquiera de $\mathcal{Q}$. Si s no interviene en $\mathcal{Q}^*$ lo eliminamos de $\mathcal{Q}$; en caso de que sí intervenga hacemos lo siguiente:

(i) añadimos un rayo vertical que se prolongue desde el extremo derecho de s hacia arriba.

(ii) si el extremo izquierdo no forma parte de $\mathcal{Q}^*$, acortamos s del siguiente modo: trasladamos el extremo izquierdo a lo largo de s en dirección al extremo derecho hasta que el extremo izquierdo aparezca en la envolvente inferior y no pertenezca al interior de ningún otro segmento. Añadimos un rayo vertical que se prolongue desde el extremo izquierdo hacia abajo.

El la figura 3.15 tenemos un ejemplo. Los segmentos sombreados son los que se acortan. Los rectángulos de color más claro contienen la parte del segmento que es eliminada del mismo y los de color más oscuro, lo que queda del segmento.

Observación 3.9.1. La transformación anterior no disminuye la complejidad de la envolvente inferior de $\mathcal{Q}$.

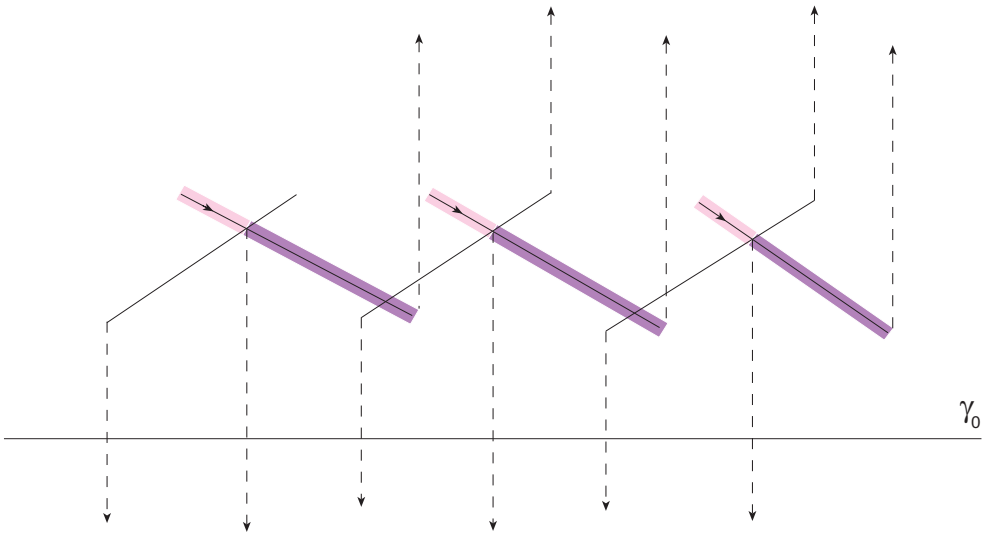


Figura 3.15: Transformando $\mathcal{Q}$ en un conjunto de curvas.

Lema 3.9.1. *Las transformaciones (i) y (ii) garantizan que cada curva se intersecte con γ_0 en un único punto y que dos curvas cualesquiera se intersecten en a lo sumo dos puntos.*

Demostración. Que cada curva se intersecte con γ_0 en un punto es inmediato partiendo de las transformaciones y del hecho de que $\mathcal{Q}$ está situado por encima de γ_0. Para ver la segunda afirmación consideremos dos curvas cualesquiera s' y r' de $\Gamma(\mathcal{Q})$. Como en $\mathcal{Q}$ no hay dos extremos de segmento con igual abscisa, las intersecciones entre s' y r' serán producto de las interseccciones entre sus respectivos segmentos de partida o de las intersecciones entre los segmentos y los rayos verticales. Los segmentos dan lugar a lo sumo a un punto de intersección entre las curvas. Como los rayos de los extremos derechos se prolongan siempre hacia arriba, esto puede dar lugar a un segundo punto de intersección entre s' y r'. Los rayos de los extremos izquierdos, sin embargo, no dan lugar a ningún punto de intersección ya que estos rayos parten de extremos que pertenecen a $\mathcal{Q}^*$ por lo que, al prolongarse hacia abajo, no van a intersectarse con nada, por definición de envolvente inferior.

□

Lema 3.9.2. *La complejidad de $z^+(\gamma_0)$ en $\mathcal{A}(\Gamma(\mathcal{Q}))$ es mayor o igual que la complejidad de $\mathcal{Q}^*$.*

Demostración. La única alteración que se hace al conjunto $\mathcal{Q}$ como tal es la de acortar algunos de sus segmentos. La parte que se elimina de un segmento es siempre una que no aparece en la envolvente inferior, por lo tanto, una vez realizadas las transformaciones de la envolvente inferior no desaparece ningún segmento que estuviera al inicio. Esto quiere decir que la complejidad de la envolvente inferior no disminuye.

□

Hemos construido de este modo una zona exterior para γ_0 que contiene a la envolvente inferior de un conjunto de segmentos. Teniendo en cuenta el lema 3.5.1 (cota superior para la complejidad de la zona exterior) y el hecho de que la envolvente inferior de un conjunto de n segmentos tiene complejidad $\Theta(\lambda_3(n))$ se llega al siguiente resultado:

Teorema 3.9.2. *La complejidad de la zona exterior de γ_0 es $\Theta(\lambda_3(n))$.*

3.10 Generalizaciones

Sea Γ un conjunto de n curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo s puntos y γ_0 una curva de Jordan no acotada que se intersecta con cada curva de Γ en a lo sumo dos puntos y que es pseudo-convexa con respecto a Γ. En la figura 3.16 tenemos un ejemplo. En esta sección vamos a ver que en determinadas circunstancias γ_0 puede insertarse en el arreglo $\mathcal{A}(\Gamma)$ en tiempo $O(\lambda_{s+1}(n))$.

Primeramente, el lema 3.4.1 se sigue cumpliendo en este caso: las extremos de la frontera exterior de una cara de $z^+(\gamma_0)$ son extremos de una arista de γ_0. La demostración, para este caso que nos ocupa ($s \geq 3$), es similar a la que se hizo en la sección correspondiente para $s = 2$. Por lo tanto, el tiempo de inserción de γ_0 se deduce, también en este caso, de la complejidad de la zona exterior, que es el

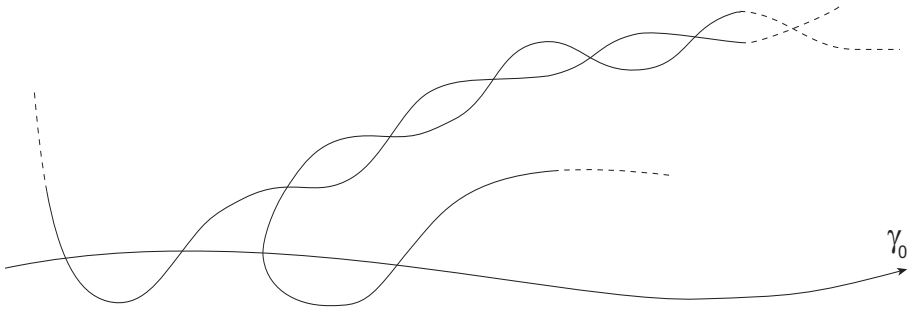


Figura 3.16: El caso general.

aspecto tratado por el lema 3.5.1. Ahora veremos que este lema sólo se generaliza en casos especiales: si s es par o si las ramas cumplen con la siguiente propiedad de monotonía.

Propiedad 3.10.1. *Decimos que un conjunto de ramas es combinatoriamente monótono con respecto a γ_0 si al orientar las ramas de modo que la base sea el punto de partida, dos ramas cualesquiera que se intersecten entre sí cumplen con lo siguiente: los puntos de intersección aparecen en una rama en el mismo orden que en la otra.*

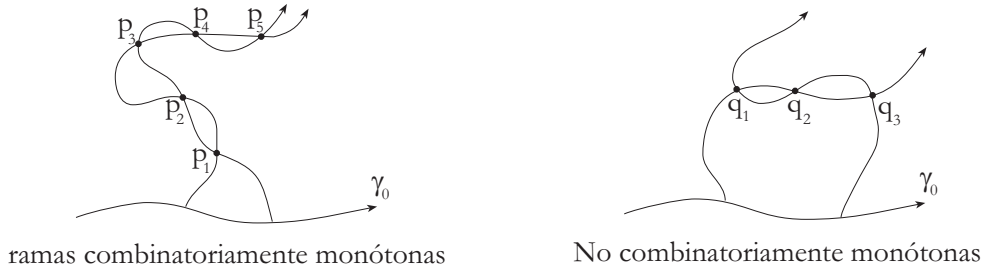


Figura 3.17: Ramas combinatoriamente monótonas.

Véase que aunque las ramas de la figura no son monótonas en el eje `OX` ni en el eje `OY`, las dos de la izquierda sí lo son según la definición anterior, pues los puntos p_1, p_2, p_3, p_4 y p_5 aparecen en este orden en cada una. Sin embargo, en el dibujo de la derecha, los puntos de intersección aparecen en una rama en el orden q_1,

q_2, q_3 y en la otra, en el orden q_3, q_2, q_1, por lo tanto, el conjunto en el que estas dos ramas aparezcan no se considerará un conjunto combinatoriamente monótono.

El lema 3.5.1 que vimos en la sección § 3.5 está basado en la propiedad 3.5.2 que nos dice que para $s = 2$ la secuencia U de las ramas que aparecen en las fronteras exteriores de $z^+(\gamma_0)$ es una secuencia $(4n, 3)$ de Davenport-Schinzel. Veremos la generalización de esta propiedad. Sea ahora U la secuencia de ramas que aparecen en las fronteras exteriores de $z^+(\gamma_0)$ cuando $s \geq 3$.

Propiedad 3.10.2. *Cuando el número s de intersecciones entre las ramas de Γ es par o las ramas son combinatoriamente monótonas, U es una secuencia $(4n, s+1)$ de Davenport-Shinzel.*

Demostración. De los tres requisitos que hay que cumplir para ser una secuencia $(4n, s+1)$ de Davenport-Shinzel U cumple sin ningún problema con los dos primeros: existen a lo sumo $4n$ símbolos y no hay dos consecutivos que sean del mismo tipo. La comprobación del tercer requisito se divide en seis casos, al igual que en 3.5.2. Todos los casos vistos allí se generalizan para $s \geq 3$ excepto el caso (6). Por lo tanto, los primeros cinco casos los veremos por arriba.

requisito (iii): este requisito nos dice que U no puede tener subcadenas de longitud $s+3$ donde dos símbolos alternen de manera sucesiva. Veremos que U cumple con este requisito cuando s es par o las ramas son combinatoriamente monótonas.

Caso (1): γ_i y γ_j son símbolos correspondientes a aristas neutrales.

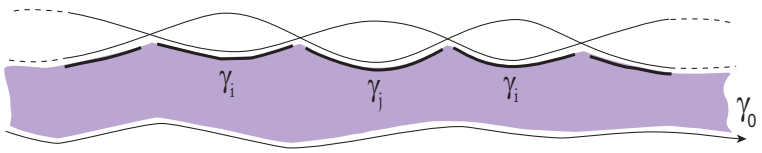


Figura 3.18: Alternancia entre curvas soporte de aristas neutrales.

Las aristas aparecen en este caso en la envolvente inferior del conjunto $\{\gamma_i, \gamma_j\}$. Con s intersecciones a lo sumo entre estas curvas la longitud máxima de las subcadenas donde γ_i y γ_j alternan es $s+1$.

Caso (2): γ_i y γ_j son símbolos correspondientes a aristas derechas de dos ramas. Supongamos que $\gamma_i < \gamma_j$. Sean $\mathcal{F}'$, $\mathcal{F}''$ y $\mathcal{F}'''$ el conjunto de caras de $z^+(\gamma_0)$ tales que: $\mathcal{F}' < \gamma_i < \mathcal{F}'' < \gamma_j < \mathcal{F}'''$.

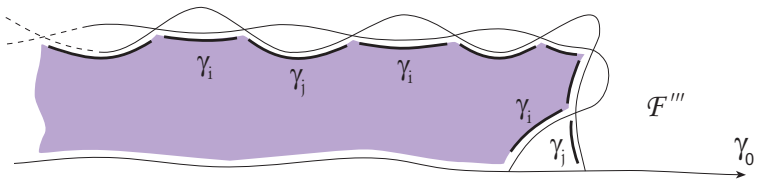


Figura 3.19: Alternancia entre lados soporte de aristas derechas.

Al igual que para $s=2$, γ_i y γ_j sólo pueden alternar en $\mathcal{F}'$, donde la longitud máxima de las subcadenas será $s+1$. Al pasar de $\mathcal{F}'$ a $\mathcal{F}''$ se tiene una última alternancia. Las subcadenas alternas tienen en este caso longitud máxima $s+2$.

Caso (3): γ_i y γ_j son símbolos correspondientes a aristas izquierdas. Este caso es simétrico con respecto al anterior.

Caso (4): γ_i es el símbolo correspondiente a una arista neutral y γ_j es el símbolo correspondiente a una arista derecha.

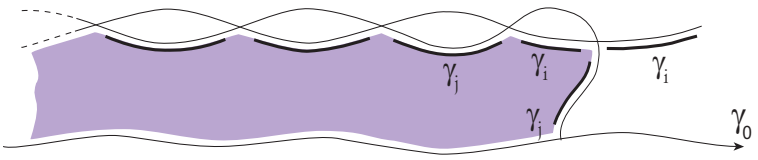


Figura 3.20: Las caras donde γ_i y γ_j pueden alternar.

Por lo que ya hemos visto, las subcadenas alternas tiene en este caso longitud máxima $s+2$.

Caso (5): γ_i es el símbolo correspondiente a una arista neutral y γ_j es el símbolo correspondiente a una arista izquierda. Este caso es simétrico con respecto al anterior.

Caso (6): γ_i es el símbolo correspondiente a una arista izquierda y γ_j es el símbolo correspondiente a una arista derecha. En este apartado tenemos dos opciones.

Opción (a): $\gamma_j < \gamma_i$.
En este caso no existe una cara de la zona exterior donde γ_i y γ_j alternen ya que ninguna cara de la zona exterior puede aparecer a la vez antes de γ_j y después de γ_i, como vemos en el ejemplo de la figura 3.21. La única alternancia que existe se da al pasar de una cara a otra. Las subcadenas alternas tienen en este caso longitud máxima 2 y son de la forma γ_j,γ_i.

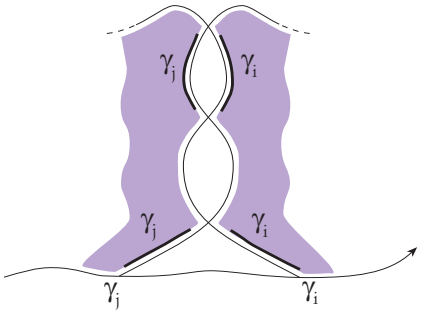


Figura 3.21: No existen caras de la zona exterior donde γ_i y γ_j puedan alternar.

Opción (b): $\gamma_i < \gamma_j$.
Si llamamos $\mathcal{F}'$, $\mathcal{F}''$ y $\mathcal{F}'''$ al conjunto de caras tales que: $\mathcal{F}' < \gamma_i < \mathcal{F}'' < \gamma_j < \mathcal{F}'''$ sabemos que γ_i y γ_j sólo pueden alternar en las caras de $\mathcal{F}''$.

Cuando las ramas son combinatoriamente monótonas, la propiedad 3.5.1 se cumple y las aristas de etiqueta γ_i aparecen en $\mathcal{F}''$ antes que las de etiqueta γ_j. Esto quiere decir que el número de alternancias en $\mathcal{F}''$ será solamente uno, lo que nos lleva a un total en U de 3 alternancias a lo sumo, con subcadenas de longitud máxima 4, como vemos en la figura 3.22.

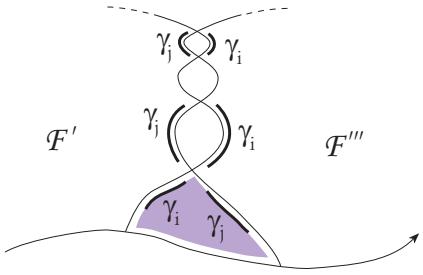


Figura 3.22: γ_i y γ_j sólo pueden alternar en las caras de $\mathcal{F}''$, que aparecen sombreadas.

En el caso en que s es par, las ramas no tienen por qué ser combinatoriamente monótonas, por lo que γ_i y γ_j pueden alternar varias veces en $\mathcal{F}''$, como vemos en el ejemplo de la figura 3.23.

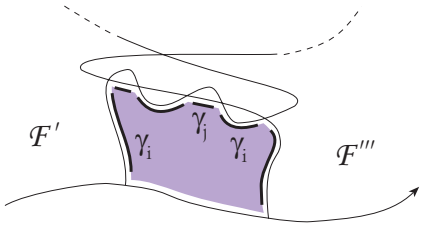


Figura 3.23: γ_i y γ_j pueden alternar varias veces en $\mathcal{F}''$.

Para este caso haremos la demostración por reducción al absurdo. Supongamos entonces que existen cadenas de longitud $s+3$ o mayor donde dos símbolos γ_i y γ_j alternan. Toda alternancia que se produzca en $\mathcal{F}''$ entre γ_i y γ_j va a ser el resultado de una intersección entre sus respectivas ramas soporte ya que estas alternancias se dan todas en una misma frontera (la de $\mathcal{F}''$). A la cadena alterna de aristas γ_i y γ_j que aparecen en la frontera de $\mathcal{F}''$ la denotaremos así: $\underbrace{\gamma_i, \ldots, \gamma_j}$, siendo γ_i y γ_j el primer y último símbolo de $\mathcal{F}''$ respectivamente.

Ahora tenemos varios casos para la cadena alterna que van a depender de si hay o no aristas γ_j en $\mathcal{F}'$ o aristas γ_i en $\mathcal{F}'''$. Las diferentes alternativas para la cadena alterna, independientemente de si pueden darse o no, son las siguientes:

$$\gamma_j \underbrace{\gamma_i, \ldots, \gamma_j} \gamma_i$$
$$\gamma_j \underbrace{\gamma_i, \ldots, \gamma_j}$$
$$\underbrace{\gamma_i, \ldots, \gamma_j} \gamma_i$$
$$\underbrace{\gamma_i, \ldots, \gamma_j}$$

Veremos la demostración para el primer caso. En el resto de los casos la demostración se hace de manera similar y se obtiene el mismo resultado: no existen cadenas de longitud $s+3$ o mayor donde γ_i y γ_j alternen.

Para que la cadena $\gamma_j \underbrace{\gamma_i, \ldots, \gamma_j} \gamma_i$ tenga longitud $s+3$ o mayor se necesitan al menos $s-1$ nuevos símbolos ya que la cadena tiene de entrada cuatro. Al ser s un número par, estamos hablando de una cantidad impar de símbolos: $s-1$. Ahora sucede que un número impar de símbolos alternos colocados en $\mathcal{F}''$ entre el primer γ_i y el último γ_j no nos lleva a una cadena alterna del tamaño buscado, ya que no se dan todas las alternancias que pretendemos, pués el primer y el último símbolo de $\mathcal{F}''$ son diferentes. En otras palabras, para obtener una cadena alterna de tamaño $s+3$ o mayor tienen que haber por lo menos s nuevos símbolos alternando en $\mathcal{F}''$. Estos s nuevos símbolos, sumados a los dos que ya tiene $\mathcal{F}''$ nos dan un total de al menos $s+2$ símbolos que alternan en la frontera de $\mathcal{F}''$. $s+2$ alternancias en una misma frontera implican $s+1$ puntos de intersección entre las ramas soporte, lo que contradice el hecho de que las ramas γ_i y γ_j se intersecten en a lo sumo s puntos.

Llegamos así a lo que queríamos demostrar. No existen subcadenas en U de longitud $s+3$ en las que dos símbolos alternen. U es, por lo tanto, una secuencia $(4n, s+1)$ de Davenport-Schinzel, siendo $\lambda_{s+1}(4n)$ la longitud máxima de U ([13], pág. 1).

□

Lema 3.10.1. *Cuando el número s de intersecciones entre las ramas de Γ es par o las ramas son combinatoriamente monótonas, la zona exterior de γ_0 tiene complejidad $O(\lambda_{s+1}(n))$.*

Observación 3.10.1. Cuando el número s de intersecciones entre las ramas de Γ es impar y las ramas no son monótonas, U no es una secuencia $(4n, s+1)$ de Davenport-Shinzel.

En la figura 3.24 tenemos un ejemplo. El número de intersecciones s entre las ramas es 3. Si U fuese una secuencia $(4n, s+1) = (4n, 4)$ de Davenport-Shinzel no podría contener subsecuencias de longitud 6 en las que dos símbolos alternaran de manera sucesiva. Sin embargo, en la siguiente figura vemos cómo las aristas izquierdas pertenecientes a la rama γ_i alternan con las aristas derechas de γ_j dando lugar a subcadenas de longitud 6.

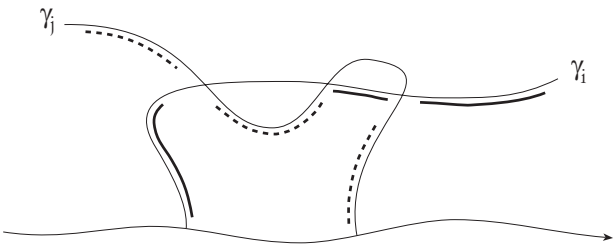


Figura 3.24: Ejemplo.

Una particularidad que se da en el caso impar es $s = 1$. En este caso la secuencia U no es efectivamente una secuencia $(4n, s+1) = (4n, 2)$ de Davenport-Schinzel (como ocurre cuando s es impar). Ver en el ejemplo de la figura 3.25 cómo aparecen cadenas alternas de longitud cuatro. Cuando $s = 1$ U es una secuencia $(4n, s+2) = (4n, 3)$, sin embargo, γ_0 puede insertarse en el arreglo en tiempo $O(\lambda_{s+1}(n)) = O(\lambda_2(n)) = O(n)$. Utilizando el método de inducción se prueba que U no alcanza aquí la longitud $\lambda_3(4n)$ a que pueden llegar tales secuencias sino que su longitud máxima en este caso es de orden lineal. Para verlo, basta con

considerar las a lo sumo $2n$ ramas como tramos de a lo sumo $2n$ pseudorrectas, lo cual se logra extendiendo las ramas sin que se corten entre sí, como se ha hecho en la figura 3.25.

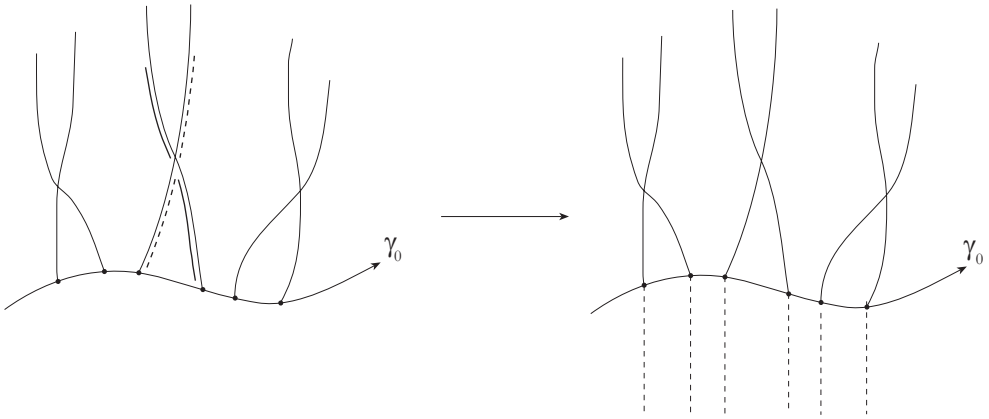


Figura 3.25: El caso de $s = 1$.

De este modo, si Γ era el conjunto inicial de ramas y Γ^* es el conjunto de ramas extendidas, entonces $\mathcal{A}(\Gamma^* \cup \gamma_0)$ es un arreglo de pseudorrectas, pues γ_0 intersecta a cada rama extendida en sólo un punto. Como ya sabemos, la complejidad de la zona de una pseudorrecta en un arreglo de pseudorrectas es lineal, luego, la complejidad de la zona de γ_0 en $\mathcal{A}(\Gamma^* \cup \gamma_0)$ es $O(n)$, y esta zona contiene a la zona exterior de γ_0 en $\mathcal{A}(\Gamma)$. Este resultado, que será utilizado en otros capítulos, podemos resumirlo en el siguiente teorema.

Teorema 3.10.1. *Si Γ es un conjunto de n curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo 1 punto, γ_0 es una curva de Jordan no acotada que se intersecta con cada curva de Γ en a lo sumo dos puntos y γ_0 es pseudo-convexa con respecto a Γ entonces γ_0 puede insertarse en el arreglo $\mathcal{A}(\Gamma)$ en tiempo $O(n)$.*

Veamos ahora, como parte de la generalización, qué sucede con el lema 3.6.1 cuando $s \geq 3$. El lema 3.6.1, que nos dice que las curvas de Γ pueden ser ordenadas

de manera tal que cada una sea pseudo-convexa con respecto a las anteriores, no se generaliza para $s \geq 3$. Cuando dos curvas se intersectan en tres o más puntos no podemos hablar de pseudo-convexidad de una con respecto a la otra ni de zona exterior. Esto quiere decir que cuando $s \geq 3$ no podemos sacar conclusiones sobre el tiempo de inserción de las curvas en el arreglo ni sobre el tiempo de construcción de $\mathcal{A}(\Gamma)$ cuando se usa un algoritmo incremental. Teniendo en cuenta todo lo visto en esta sección podemos enunciar ya el nuevo resultado obtenido.

Teorema 3.10.2. *Si Γ es un conjunto de n curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo s puntos, γ_0 es una curva de Jordan no acotada que se intersecta con cada curva de Γ en a lo sumo dos puntos y γ_0 es pseudo-convexa con respecto a Γ entonces, si las ramas de γ_0 son combinatoriamente monótonas, si s es un número par o $s = 1$, γ_0 puede insertarse en el arreglo $\mathcal{A}(\Gamma)$ en tiempo $O(\lambda_{s+1}(n))$.*

Por último, veremos que $\lambda_{s+1}(n)$ es también cota inferior para la complejidad de la zona exterior de γ_0 en el arreglo de curvas que tratamos en esta sección: curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo s puntos y con respecto a las cuales γ_0 es pseudo-convexa. La demostración es similar a la que se hizo en la sección §3.9 para el caso de $s = 2$. Sea entonces $\mathcal{Q}$ un conjunto de arcos de Jordan (curvas de Jordan acotadas) monónotos en el eje `OX` (la intersección entre un arco y cualquier recta paralela al eje `OY` se da en a lo sumo un punto) que se intersectan dos a dos en a lo sumo $s - 1$ puntos.

El resultado que aparece en ([4], pág. 396) nos dice que la envolvente de $\mathcal{Q}$ tiene complejidad $\Theta(\lambda_{s+1}(n))$. Ponemos entonces γ_0 en el eje `OX` y el conjunto $\mathcal{Q}$ por encima de γ_0 de modo que al extender los arcos de $\mathcal{Q}$ a curvas no acotadas, la semi-zona superior de γ_0 contenga a la envolvente inferior de $\mathcal{Q}$, de complejidad $\Theta(\lambda_{s+1}(n))$.

La extensión se hace de manera similar a como se hizo en §3.9: los extremos derechos de los arcos se extienden hacia arriba en dirección vertical y los extremos izquierdos se trasladan por el arco en cuestión hasta que aparezcan en la

envolvente inferior; momento en el que son extendidos hacia abajo, también en dirección vertical. Como los arcos de partida eran monótonos y se intersectaban dos a dos en a lo sumo $s-1$ puntos, después de la transformación, dos cualesquiera de las curvas obtenidas se intersectan en a lo sumo s puntos.

Llegamos así a un conjunto $\mathcal{Q}^*$ de curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo s puntos, con respecto al cual γ_0 es pseudo-convexa y donde la zona exterior de γ_0 contiene a la envolvente inferior de $\mathcal{Q}$. Teniendo en cuenta el lema 3.10.1 y que la envolvente inferior de $\mathcal{Q}$ tiene cota inferior $\Omega(\lambda_{s+1}(n))$ llegamos a la siguiente conclusión.

Teorema 3.10.3. *Cuando el número s de intersecciones entre las ramas de Γ es par o las ramas son combinatoriamente monótonas, la zona exterior de γ_0 tiene complejidad $\Theta(\lambda_{s+1}(n))$.*

3.11 Curvas Tangentes en el Arreglo

Un hecho interesante, y que utilizaremos en el capítulo 5, es que el resultado 3.7.1 para curvas que se intersectan dos a dos en a lo sumo dos puntos es también válido cuando en el conjunto de curvas hay pares de ellas que se intersectan en un punto pero de manera tangente. Esto se debe a que los lemas 3.4.1, 3.5.1 y 3.6.1, bases del teorema, se siguen cumpliedo en este caso.

Para verlo basta con transformar imaginariamente las curvas de modo que cada punto de tangencia se convierta en dos puntos de intersección transversal sin que por ello se altere el resto de la topología del arreglo. El conjunto de curvas con casos de tangencia queda transformado así en un conjunto de curvas como el que hemos analizado en este capítulo: curvas que se intersectan dos a dos en a lo sumo dos puntos. En la siguiente figura vemos representada la transformación.

Figura 3.26: El arreglo antes y después de la transformación.

En esta figura 3.26 aparece el caso en que el punto de tangencia se da entre la curva γ_0 que se inserta y las que ya están en el arreglo. Como puede comprobarse, los lemas 3.4.1 (sobre la frontera exterior) y 3.6.1 (sobre la ordenación de las curvas) se siguen cumpliendo en el arreglo con curvas tangentes.

Cuando el punto de tangencia se da entre curvas que ya están en el arreglo, como las de la figura 3.27, se tiene entonces que después de la transformación el número de aristas en la zona ha aumentado, luego, la cota para la complejidad de la zona exterior después de la transformación sirve también de cota para la complejidad de la zona antes de la transformación. Con esto queda probada la validez del lema 3.5.1. En la figura hemos resaltado las nuevas aristas que aparecen en la zona al hacerse la transformación.

Figura 3.27: El arreglo antes y después de la transformación.

Teorema 3.11.1. *Si* Γ *es un conjunto de curvas simples no acotadas que se intersectan dos a dos en a lo sumo dos puntos y donde cualquier par de curvas puede intersectarse entre sí en un único punto de manera tangente entonces el arreglo* $\mathcal{A}(\Gamma)$ *puede construirse en tiempo* $O(n\,\lambda_3(n))$.

El proceso de construcción del arreglo es también el mismo gracias, entre otras cosas, al lema 3.4.1. Se tendrá así un primer paso en el que se ordenan las curvas y un segundo paso en el que se insertan en el arreglo de una en una.

Veamos ahora la generalización a curvas que se intersectan dos a dos en a lo sumo s puntos. Se trata ahora de un conjunto Γ de n curvas simples no acotadas con un índice de intersección dos a dos menor o igual que s; siendo dicho índice el número de intersecciones secantes más dos veces el número de intersecciones tangentes. γ_0 es ahora una curva de Jordan no acotada que se intersecta con cada curva de Γ en a lo sumo dos puntos y que en caso de que se intersecte con una curva en un sólo punto, este punto puede ser de tangencia. Teniendo en cuenta la transformación vista se llega a que, si γ_0 es pseudo-convexa con respecto a Γ entonces:

Teorema 3.11.2. *γ_0 puede insertarse en el arreglo $\mathcal{A}(\Gamma)$ en tiempo $O(\lambda_{s+1}(n))$ si las curvas de Γ son combinatoriamente monótonas o si s es un número par.*

3.12 Problemas Abiertos

El problema que queda abierto es el de determinar la complejidad de la semi-zona de γ_0 que queda al otro lado de γ_0, figura 3.28. Este problema está abierto para $s \geq 1$ cuando nos restringimos a familias específicas de curvas. Para curvas arbitrarias la complejidad es $O(\lambda_{s+2}(n))$ ([13], pág. 125).

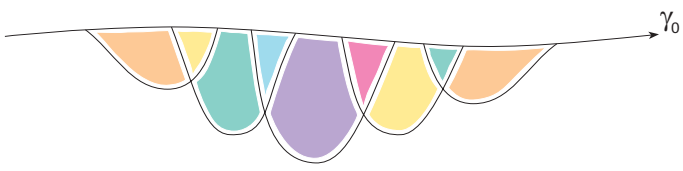


Figura 3.28: La semi-zona al otro lado de γ_0.

Véase que cuando $s = 1$ estamos ante un problema que lleva abierto ya más de 10 años (citado por M. Sharir en el Workshop de Dagstuhl [12]):

Problema 1. *Dada una circunferencia y un arreglo de pseudorrectas, determinar la complejidad de la semi-zona de la circunferencia que queda en su interior.*

En la siguiente figura tenemos un ejemplo. La parte de la zona a la que nos referimos en el problema aparece sombreada.

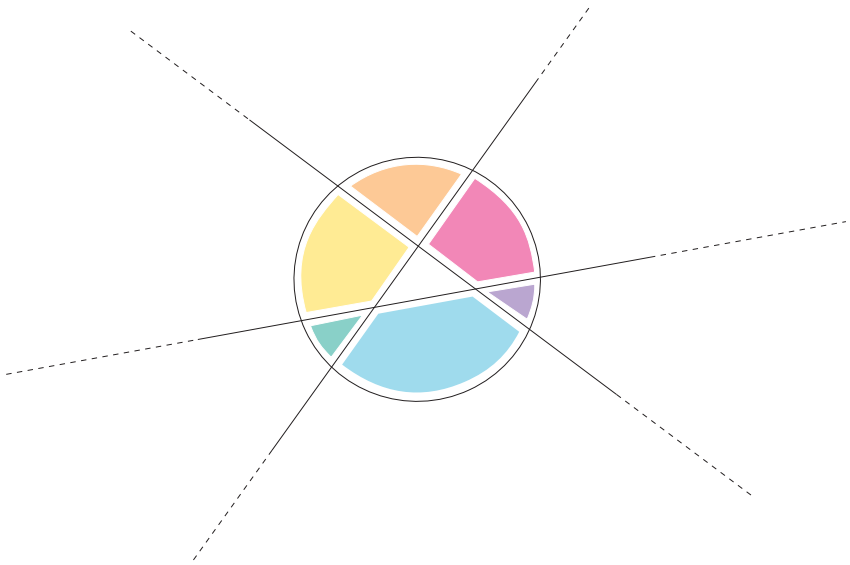


Figura 3.29: Problema abierto.

Capítulo 4

El Problema del Anillo

4.1 Definición del Problema

Un anillo es la región del plano limitada por dos circunferencias concéntricas. Si llamamos r_1 y r_2 a los radios de estas circunferencias, la anchura del anillo es el valor absoluto de la diferencia $r_1 - r_2$; y el radio del anillo, la media aritmética de r_1 y r_2. El problema que vamos a resolver es el siguiente:

Problema 2. *Dado un conjunto de n puntos en el plano y un anillo $\mathcal{R}$ de radio y anchura fijos, hallar una posición de $\mathcal{R}$ en el plano donde cubra la mayor cantidad de puntos.*

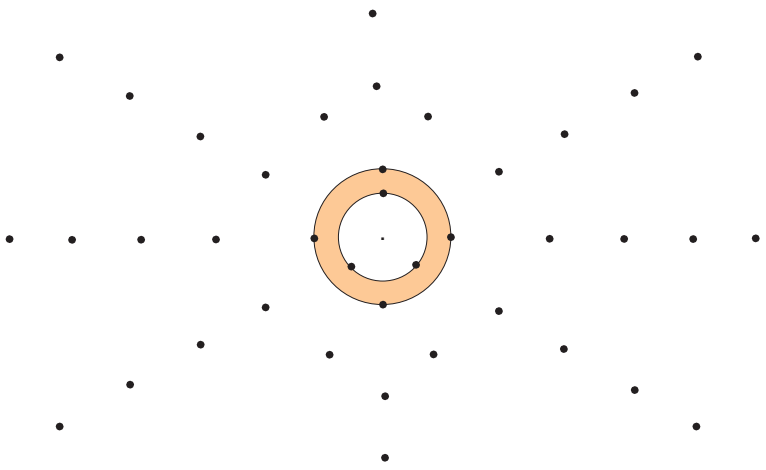


Figura 4.1: Ilustración del problema.

La posición del anillo en el plano va a venir dada por la posición de su centro $\mathcal{C}$, que es el centro común de las dos circunferencias. Para resolver este problema vamos a tener en cuenta la siguiente propiedad:

Propiedad 4.1.1. *Un anillo con centro en $\mathcal{C}$ cubre al punto p si y sólo si, cuando trasladamos el anillo hasta p, el punto $\mathcal{C}$ queda cubierto.*

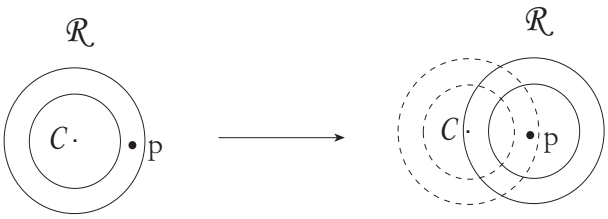


Figura 4.2: Trasladando el anillo.

Llamemos ahora $\mathcal{R}_i$ al anillo del mismo tamaño que $\mathcal{R}$ con centro en el punto p_i. La propiedad anterior nos dice que si en una determinada posición $\mathcal{R}$ cubre los puntos p_1, p_2, ..., p_m entonces los anillos $\mathcal{R}_1$, $\mathcal{R}_2$, ..., $\mathcal{R}_m$ cubren al punto $\mathcal{C}$, o lo que es lo mismo, $\mathcal{C}$ pertenece a la intersección de los m anillos. En la figura 4.3 tenemos un ejemplo con tres puntos.

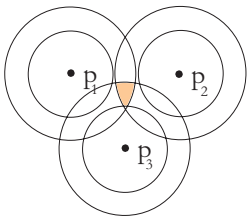


Figura 4.3: Un ejemplo con tres puntos.

Sabiendo que el arreglo generado por un conjunto de n objetos en el plano nos permite conocer todas las intersecciones y superposiciones entre ellos resolveremos nuestro problema a través del siguiente problema equivalente:

Problema 3. *Dado un conjunto de n anillos* $\{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_n\}$*, construir el arreglo que generan.*

La solución del problema original (la posición de $\mathcal{R}$) la vamos a tener en la cara del arreglo que pertenezca al mayor número de anillos. Esta cara no tiene por qué ser única, como vemos en el ejemplo de la figura 4.4. Cualquier punto de una de estas caras va a ser una solución. Esto quiere decir que una vez que el arreglo esté construido vamos a etiquetar cada cara con el número de anillos a los que pertenece.

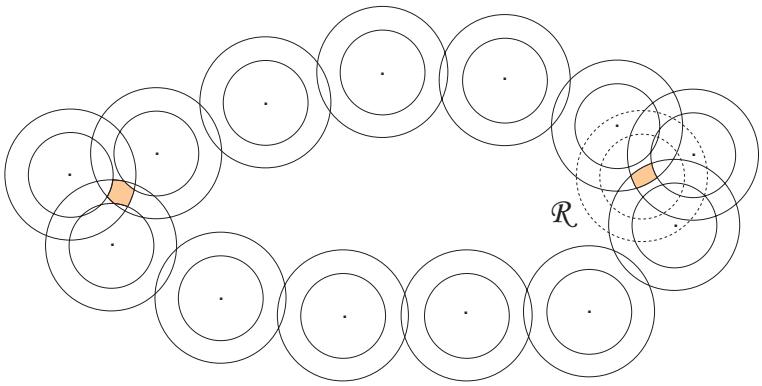


Figura 4.4: Problema equivalente.

En el ejemplo de la figura 4.4 las caras que contienen las diferentes posiciones para $\mathcal{R}$ son el resultado de la intersección de tres anillos. Hemos dibujado el anillo $\mathcal{R}$ en una de las infinitas posiciones que componen la solución. Como es de esperar, en cualquiera de estas posiciones $\mathcal{R}$ cubre tres de los puntos del conjunto de entrada. Los puntos de entrada aparecen también resaltados en la figura.

Para construir el arreglo de anillos veremos el conjunto $\{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_n\}$ como un conjunto de circunferencias. Este conjunto va a estar formado por las dos circunferencias que definen a cada anillo. Pasamos así de un conjunto de n anillos a un conjunto de $2n$ circunferencias $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_{2n}\}$. Si establecemos ahora que el tamaño del conjunto de circunferencias es n, nuestro problema se convierte en el siguiente:

Problema 4. *Dado un conjunto de n circunferencias de radio r_1 o r_2, construir el arreglo que generan.*

Cuando las n circunferencias son del mismo radio existe ya un algoritmo [3] que construye el arreglo en tiempo $O(n^2)$. El algoritmo utiliza el método incremental en el que las circunferencias son insertadas una a una en la estructura. Para el problema general de n circunferencias de radios diferentes, sin embargo, no existe todavía ningún algoritmo incremental (que nosotros sepamos) que construya el arreglo de circunferencias en tiempo cuadrático.

Cuando el número de radios diferentes es una constante, nuestro algoritmo construye el arreglo de circunferencias en tiempo $O(n^2)$, como veremos al final de este capítulo. En realidad, si el número de radios diferentes es k, la complejidad del tiempo de construcción del arreglo es $O(n^2\, log(k))$.

Para resolver nuestro problema supondremos que no hay más de dos circunferencias incidentes en un mismo punto. El caso degenerado en que más de dos circunferencias pueden pasar por un punto es analizado en el capítulo de "Técnicas de Implementación".

4.2 Estrategia General

En la sección §2.7 del capítulo de "Técnicas de Implementación" vimos que si nos dan el orden en que un conjunto de curvas se intersectan entre sí podemos construir el arreglo de las mismas en tiempo cuadrático. Esta será la estrategia que utilizaremos para construir el arreglo de circunferencias de dos radios: hallar primero el orden en que cada circunferencia intersecta a las demás. Demostraremos que es posible hallar este orden para todas las circunferencias en tiempo cuadrático. Los pasos para hallar la solución podemos resumirlos entonces del siguiente modo:

Algoritmo General

[1] Para cada circunferencia $\mathcal{D}_i$ construir la lista ordenada de puntos de intersección entre $\mathcal{D}_i$ y las demás circunferencias.

[2] Construir el arreglo de circunferencias partiendo de las listas de puntos de intersección.

[3] Etiquetar las caras del arreglo con el número máximo de anillos a que pertenece.

[4] Hacer k igual al valor máximo de las etiquetas.

[5] Devolver las caras del arreglo con etiqueta k.

El único paso que vamos a desarrollar es el [1] ya que no es inmediato que se ejecute en tiempo cuadrático. Los pasos [2] y [3] los tenemos en el capítulo de Técnicas de Implementación y consumen tiempo cuadrático mientras que los pasos [4] y [5] no tienen ninguna complicación.

4.3 Calculando el Orden de Intersección

Nos dedicamos entonces a hallar las listas ordenadas de puntos de intersección para lo cual tenemos que saber primero cómo vamos a obtener estas intersecciones de manera ordenada. Una primera estrategia para saber el orden en que una circunferencia $\mathcal{D}$ intersecta a las demás es hallar las intersecciones de $\mathcal{D}$ con las circunferencias y luego ordenarlas. Al tratarse de n circunferencias, la complejidad total será $O(n^2 \log(n))$. Esta cota, sin embargo, puede bajarse hasta $O(n^2)$ si dividimos las circunferencias en arcos de circunferencia y hallamos por separado el orden en que cada arco de cicunferencia intersecta a los demás. Esta es la idea que vamos desarrollar. En particular, las circunferencias serán divididas por su diámetro horizontal, de modo que por cada circunferencia obtendremos dos arcos, como vemos en el ejemplo de la figura 4.5.

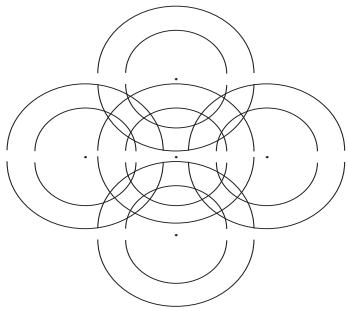


Figura 4.5: Las circunferencias divididas en arcos.

A continuación, vamos a agrupar los arcos en conjuntos de manera tal que dentro de cada conjunto todos los arcos tengan igual radio y convexidad. Obtenemos entonces cuatro conjuntos de arcos, como vemos en la figura 4.6.

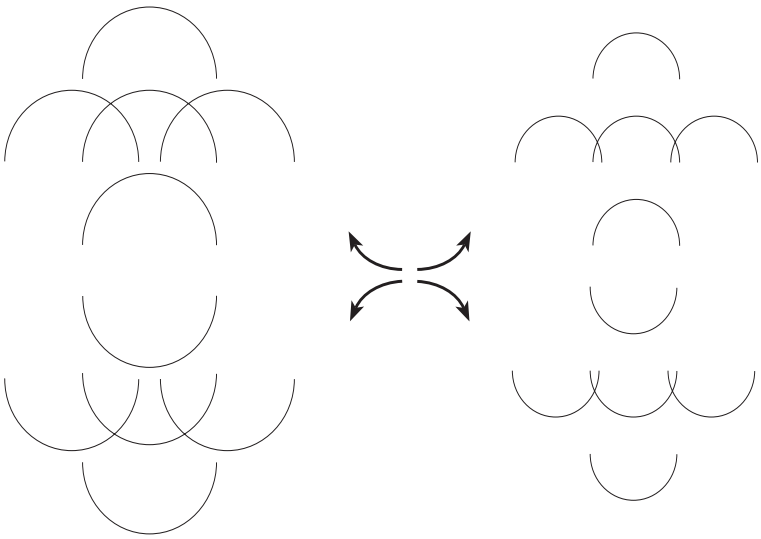


Figura 4.6: Los cuatro conjuntos de arcos.

Si ahora construimos por cada conjunto el arreglo que los arcos del mismo generan podemos tener ya el orden que buscamos:

(i) del arreglo al que un arco pertenece obtenemos de manera inmediata el orden en que intersecta a los arcos de su mismo tipo (bastará con hacer su recorrido dentro del arreglo).

(ii) para hallar el orden en que intersecta a los arcos de los otros arreglos, lo insertamos en cada uno de los otros tres arreglos (mientras lo insertamos vamos construyendo la lista ordenada de puntos de intersección y, al terminar la lista, lo borramos).

De esta manera, por cada arco vamos a tener finalmente cuatro listas ordenadas de puntos de intersección, una por cada arreglo. Las cuatro listas de cada arco las mezclamos en tiempo lineal y habremos terminado.

A los arcos de una circunferencia los llamaremos inferior y superior o cóncavo y convexo respectivamente. Utilizaremos la notación Γ_i, $1 \leq i \leq 4$ para referirnos a estos conjuntos de arcos de igual tipo. Al conjunto de arcos en total lo llamaremos Γ.

Supondremos, sin pérdida de generalidad, que en ningún conjunto Γ_i hay dos extremos de arco con igual abscisa. Esto nos será de utilidad más adelante.

Veremos ahora la construcción de los arreglos de arcos y la inserción de un arco en un arreglo al que no pertenece. Por la simetría que se da entre los cuatro conjuntos de arcos y para facilitar la explicación nos vamos a dedicar solamente a uno de los conjuntos de arcos inferiores.

4.4 Arreglos de Arcos. Construcción.

Sea Γ_i uno de los conjuntos de arcos inferiores, $1 \leq i \leq 4$. Como todos los arcos de Γ_i tienen igual radio y convexidad, la intersección entre dos cualesquiera de ellos, en caso de existir, es única (la doble intersección entre dos arcos de Γ sólo es posible si los arcos tienen radios diferentes o son de convexidad diferente, como se ve en la figura 4.7).

Que los arcos de Γ_i se intersecten dos a dos en a lo sumo un punto nos hace pensar en la construcción de $\mathcal{A}(\Gamma_i)$ utilizando un algoritmo incremental clásico. Como su nombre lo indica, con un algoritmo incremental los arcos se van a añadir al

Figura 4.7: Intersección entre los arcos de Γ.

arreglo de uno en uno. En este caso, sin embargo, la inserción de un arco en el arreglo ya no seguiría la "línea clásica" en la que cada dos puntos de intersección consecutivos entre el arco y el arreglo se añade una arista, pués los arcos son objetos acotados. En la figura 4.8 tenemos un ejemplo: si durante la inserción del arco dibujado en pespuntes la cara sombreada se recorre en sentido horario, el siguiente punto de intersección en aparecer después de p es q; sin embargo, pq no es una arista del arreglo que queremos construir.

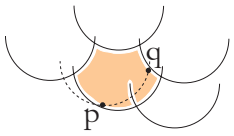


Figura 4.8: Aunque p y q aparecen en ese orden, pq no es arista del arreglo en cuestión.

Por otra parte, el hecho de que los arcos sean objetos acotados da lugar a que en el arreglo $\mathcal{A}(\Gamma_i)$ puedan aparecer varias componentes conexas, lo que no facilita las cosas.

A fin de evitar la dispersión del arreglo en varias componentes conexas y resolver el problema de la conexión vamos a añadir segmentos auxiliares que nos conecten los arcos entre sí. Estos segmentos se añadirán de manera tal que siempre exista una única componente conexa durante todo el proceso de construcción del arreglo. Más adelante vemos que con estos segmentos queda resuelto también el primer problema: garantizar que durante la inserción de un arco las intersecciones consecutivas sean extremos de aristas.

Para conectar los arcos entre sí ponemos un segmento horizontal t que estará por encima de todos los arcos y añadimos segmentos verticales que unan a cada arco con el segmento t. Si entendemos por centro de un arco el centro de la circunferencia a la que pertenece, y si denotamos por y_{max} al máximo de las ordenadas de los centros de los arcos y por x_{max} y x_{min} a las abscisas máxima y mínima respectivamente de los extremos de los arcos, podemos decir entonces que el segmento t se extiende desde el punto $(x_{min} - 1, y_{max} + 1)$ hasta el punto $(x_{max} + 1, y_{max} + 1)$. Los segmentos verticales serán dos por cada arco y partirán desde cada uno de los extremos del arco, extendiéndose hacia arriba hasta chocar con t. En la figura 4.9 vemos cómo quedaría el arreglo.

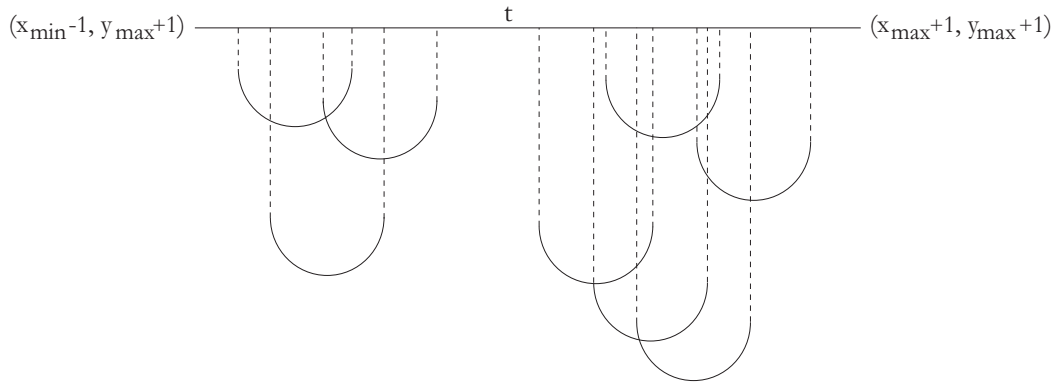


Figura 4.9: El arreglo de arcos con los segmentos añadidos.

Definición 4.4.1. *Llamaremos* arco extendido *al objeto formado por un arco y sus respectivos segmentos verticales.*

Vistas estas modificaciones, la construcción de un arreglo de arcos consiste en construir el respectivo arreglo de arcos extendidos. Un arreglo de arcos extendidos se construye añadiendo primero el segmento t y luego uno a uno los arcos extendidos.

Cuando se trate de los conjuntos de arcos superiores, el segmento t estará por debajo de todos los arcos y los segmentos verticales se extenderán desde los extremos de los arcos hacia abajo.

Complejidad del Tiempo de Construcción

Aunque los segmentos auxiliares se añadieron con el objetivo de mantener el arreglo conexo en todo momento, también trajeron como consecuencia que el arreglo se convirtiera en un arreglo de pseudorrectas, como veremos enseguida. Se cumple entonces el Teorema de la Zona [5] y la complejidad de insertar un arco va a ser $O(n)$ y no $O(n\,\alpha(n))$, como sería el caso si no hubiésemos añadido los segmentos, ya que se trataría entonces de un arreglo de arcos de Jordan (arcos de circunferencia). Ver en ([13], pág. 125) la complejidad de insertar un arco en este tipo de arreglos.

Para darnos cuenta de que estamos ante un arreglo de pseudorrectas hacemos lo siguiente: extendemos imaginariamente los segmentos verticales más allá del segmento horizontal t. Los segmentos izquierdos de los arcos se extenderán en dirección vertical hacia más infinito mientras que los segmentos derechos se extenderán también hacia más infinito pero con un ángulo de inclinación que será el mismo para todos los arcos del arreglo. En la figura 4.10 tenemos un ejemplo.

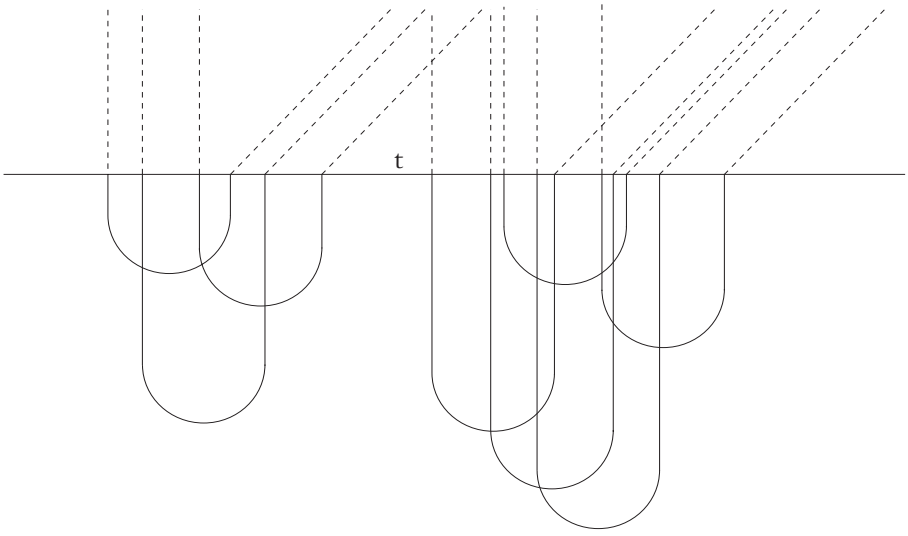


Figura 4.10: Viendo el arreglo como un arreglo de pseudorrectas.

Con estas transformaciones el conjunto de arcos extendidos queda convertido en un conjunto de curvas simples, cada una de las cuales separa el plano y para las

que se cumple que al ser tomadas de dos en dos se intersectan transversalmente en un punto. Este conjunto de curvas es entonces un conjunto de pseudorrectas [14]. Como en los arreglos de pseudorrectas se cumple el Teorema de la Zona llegamos a que la zona de cada arco extendido tiene complejidad lineal. Usando un algoritmo incremental clásico podemos construir los arreglos de arcos extendidos en tiempo cuadrático. Teniendo en cuenta que el arreglo de arcos extendidos contiene al correspondiente arreglo de arcos (el arreglo sin los segmentos auxiliares) y que nuestro objetivo al extender los arcos era tan sólo facilitar la construcción del arreglo de los mismos podemos enunciar entonces el siguiente resultado:

Teorema 4.4.1. *Un arreglo de arcos de circunferencia de igual radio y convexidad puede construirse en tiempo* $O(n^2)$.

4.5 Inserción de Arcos de Distinto Tipo

Supongamos ahora que $\gamma \notin \Gamma_i$. En este caso γ tiene radio distinto al de los arcos de Γ_i o convexidad diferente con lo cual, el número de intersecciones entre γ y un arco cualquiera de Γ_i va a ser a lo sumo dos. La cota que se tiene hasta el momento para la complejidad de la zona de γ en $\mathcal{A}(\Gamma_i)$ es $O(\lambda_4(n)) = O(n\,2^{\alpha(n)})$. Sin embargo, sabiendo que tratamos con arcos de circunferencia, que todos los arcos de Γ_i son del mismo tipo y que $\gamma \notin \Gamma_i$ concluimos que γ se comporta como un arco pseudo-convexo con respecto a Γ_i. Aplicando el teorema 3.10.1 llegamos a que γ puede insertarse en $\mathcal{A}(\Gamma_i)$ en tiempo lineal. Tenemos entonces el siguiente lema:

Lema 4.5.1. *Dados* γ *y un conjunto* Γ_i *tal que* γ *no pertenece a* Γ_i, γ *puede insertarse en el arreglo* $\mathcal{A}(\Gamma_i)$ *en tiempo* $O(n)$.

Demostración. Aplicación del Teorema 3.10.1.

□

En la figura 4.11 hemos dibujado todos los casos posibles: los cuatro tipos posibles de γ y, para cada uno, los tres tipos de arreglos en que va a ser insertado,

que no son más que los arreglos $\mathcal{A}(\Gamma_i)$ tales que γ no pertenece a Γ_i. Hemos orientado γ en sentido horario.

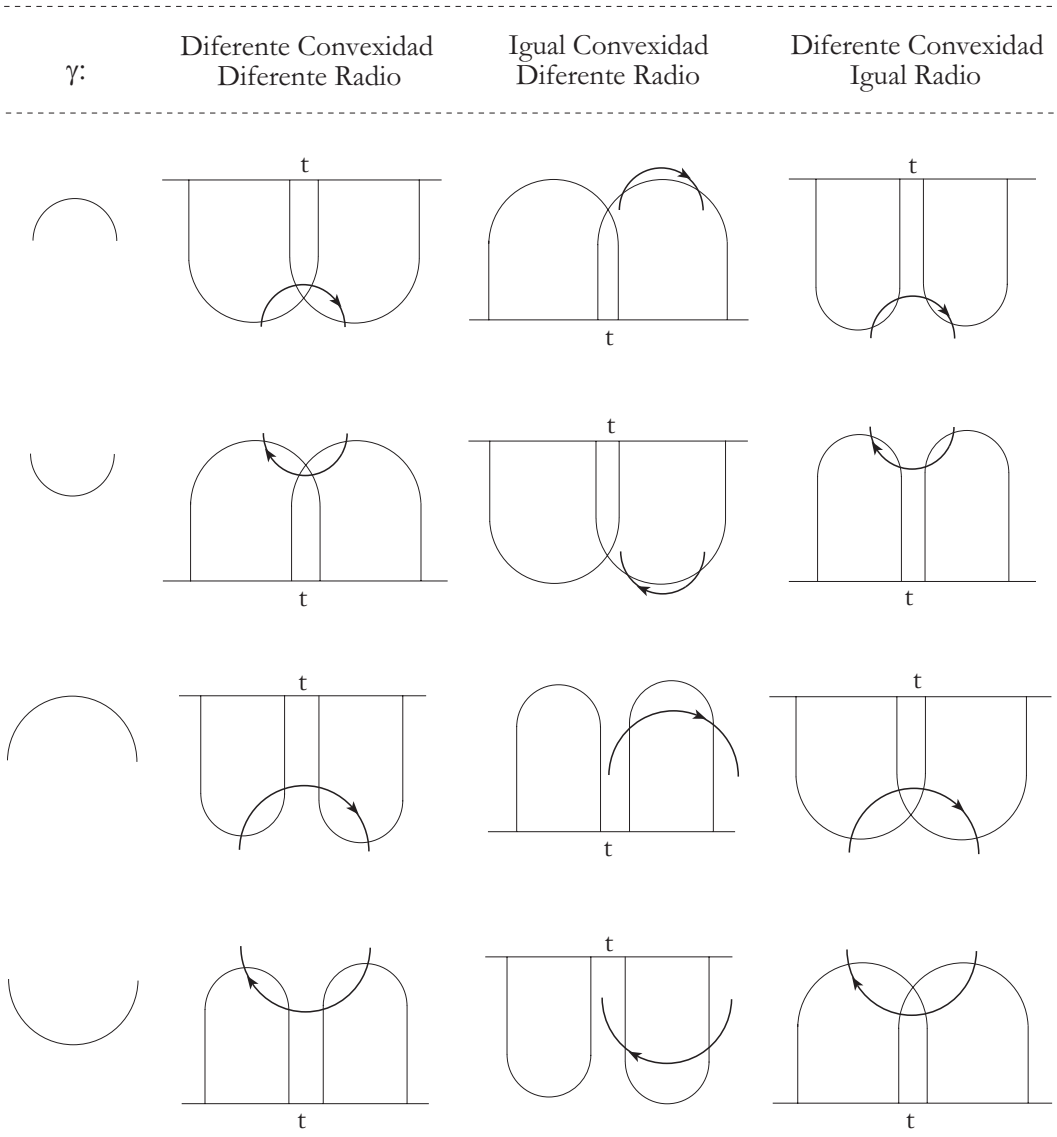


Figura 4.11: Todos los casos posibles.

Como vemos en el dibujo, al orientar γ en sentido horario la zona exterior de γ queda a su izquierda, a excepción de las configuraciones donde tiene igual convexidad y mayor radio que los arcos del arreglo: las dos últimas de la columna central. Otro detalle a mencionar es que las configuraciones donde γ tiene igual

convexidad que los arcos del arreglo y menor radio (la dos primeras configuraciones de la columna central) son las únicas donde una frontera exterior puede contener los extremos del segmento t. En estos casos, sólo una de las fronteras exteriores contendrá los extremos de t.

4.6 Construcción de las Listas

En esta sección vamos a tratar la construcción de las listas ordenadas de puntos de intersección, que ya sabemos serán cuatro por cada arco. Sea entonces $\mathcal{A}(\Gamma_i)$ uno cualquiera de los arreglos de arcos.

Para construir la lista con el orden en que γ intersecta los arcos de este arreglo lo primero que tenemos que saber es si γ pertenece o no al conjunto Γ_i. Dadas las estructuras de datos con las que contamos (las que vimos en el capítulo de "Técnicas de Implementación") la respuesta a esta pregunta se obtiene del propio γ en tiempo constante.

Caso $\gamma \in \Gamma_i$:
$l(\gamma)$ se obtiene de manera inmediata al hacer el recorrido de γ dentro del arreglo. El mismo arco γ nos da acceso a su localización dentro del arreglo, gracias a que todo arco almacena una de sus aristas en el arreglo al que pertenece.

Caso $\gamma \notin \mathcal{A}(\Gamma_i)$:
la lista de las intersecciones entre γ y el conjunto Γ_i la obtenemos insertando γ en el arreglo. La lista se construye a medida que los puntos de intersección van apareciendo mientras γ es insertada. Una vez que se tiene la lista, se borra γ.

Para construir las listas vamos a tener en cuenta los siguientes detalles:

- Las listas serán construidas en paralelo a la aparición de los puntos de intersección entre γ y el arreglo en cuestión. γ se recorrerá en sentido horario (la orientación que vimos en 4.11), a fin de facilitar la posterior mezcla de listas.

- Todas las intersecciones entre γ y segmentos auxiliares las vamos a dejar pasar (los segmentos auxiliares no forman parte del arreglo original).
- Por cada punto de intersección entre γ y otro arco δ se creará un nuevo elemento en $l(\gamma)$ en el que se guardarán las coordenadas del punto y otros dos datos que nos harán falta a la hora de construir el arreglo a partir de las listas. Estos dos datos son la circunferencia soporte de δ, y si el punto de intersección en cuestión es el único punto de intersección entre las dos circunferencias soporte (la de γ y la de δ), en cuyo caso sería un punto de tangencia, o es el punto de más a la izquierda o el de más abajo de los dos puntos de intersección. Para facilitar las cosas, las coordenadas de los extremos de los arcos estarán también almacenadas con el propio arco.
- Una vez construida la lista, el último paso es siempre guardar en el campo correspondiente de γ un puntero a la lista.

Al final, cada arco tendrá cuatro listas, correspondientes a las intersecciones con los cuatro arreglos.

Complejidad del Tiempo de Construcción de las Listas

Como hemos visto, las listas se construyen a medida que aparecen los puntos de intersección entre γ y el arreglo en cuestión. Por la estrategia utilizada, los puntos a almacenar son hallados en el mismo orden en que se encuentran en γ, de ahí que almacenarlos de manera ordenada en la lista requiera sólo tiempo constante. Los datos adicionales que se almacenan con los puntos de intersección se obtienen también en tiempo constante:

- la circunferencia soporte de un arco viene con cada arco y

- la determinación de si un punto de intersección es el único punto de intersección entre dos circunferencias, si es un punto de tangencia o si es el de más a la derecha (en caso de que se trate de dos intersecciones entre las circunferencias) puede calcularse también en tiempo constante.

Teorema 4.6.1. *La lista $l(\gamma)$ con el orden en que γ intersecta a los arcos de un conjunto Γ_i puede ser construida en tiempo $O(n)$.*

Demostración. Tanto si $\gamma \in \mathcal{A}(\Gamma_i)$ como si $\gamma \notin \mathcal{A}(\Gamma_i)$, la zona que se recorre de γ en el arreglo $\mathcal{A}(\Gamma_i)$ tiene complejidad $O(n)$ (se trata de recorrer la zona de γ si $\gamma \in \Gamma_i$ o de insertar γ en $\mathcal{A}(\Gamma_i)$ si $\gamma \notin \Gamma_i$, recorriendo en este caso su zona exterior). □

4.7 Algoritmo

Finalmente, los pasos para construir el arreglo de anillos son los que aparecen en el recuadro que tenemos a continuación. En el paso $[7a]$, para facilitar el acceso inmediato a los dos arcos de una circunferencia y evitar así el consumo de tiempo que implicaría el tener que buscarlos, adoptamos la siguiente norma: los arcos inferior y superior de la $i-$*ésima* circunferencia del vector de circunferencias se encontrarán respectivamente en las posiciones $2i-1$ y $2i$ del vector de arcos, donde $i = 1..2n$. Hemos supuesto que el primer elemento de un arreglo tiene subíndice 1.

Algoritmo de los Anillos

[1] Dividir las dos circunferencias de cada anillo por su diagonal horizontal y llamar Γ al conjunto de arcos de circunferencia.

[2] Formar los conjuntos Γ_1, Γ_2, Γ_3 y Γ_4 de arcos de igual radio y convexidad.

[3] Construir los arreglos $\mathcal{A}(\Gamma_1)$, $\mathcal{A}(\Gamma_2)$, $\mathcal{A}(\Gamma_3)$ y $\mathcal{A}(\Gamma_4)$.

[4] Para cada arco $\gamma \in \Gamma$ repetir desde i=1 hasta 4:

a) si $\gamma \in \mathcal{A}(\Gamma_i)$ obtener la lista $l_i(\gamma)$.

b) si $\gamma \notin \mathcal{A}(\Gamma_i)$ entonces:

I) insertar γ en $\mathcal{A}(\Gamma_i)$.

II) obtener la lista $l_i(\gamma)$.

III) borrar γ de $\mathcal{A}(\Gamma_i)$.

[5] Para cada arco $\gamma \in \Gamma$ mezclar las cuatro listas en $l(\gamma)$.

[6] Inicializar $\mathcal{A}$ a arreglo vacío.

[7] Para cada circunferencia $\mathcal{D}_i$ perteneciente a un anillo:

a) mezclar $l(\gamma_{2i-1})$ y $l(\gamma_{2i})$, las listas correspondientes a sus dos arcos de circunferencia, y obtener $l(\mathcal{D}_i)$.

b) insertar $l(\mathcal{D}_i)$ en $\mathcal{A}$.

[8] Devolver $\mathcal{A}$.

Análisis de la Complejidad del Algoritmo

Esta sección es a modo de resumen ya que para la mayoría de los pasos hemos visto ya cuál es la complejidad. Primero, los pasos [1] y [2] son los únicos que consumen tiempo lineal. En el paso [3], cada arreglo se construye en tiempo cuadrático (vimos que se trataba de arreglos de pseudorrectas). Como el número de arreglos es cuatro, la complejidad de este paso es cuadrática.

El paso [4] es un ciclo que se repite cuatro veces para cada arco, de los que se tiene una cantidad lineal: dos por cada una de las n circunferencias. El ciclo se ejecuta entonces $8n$ veces. Dentro de este ciclo se ejecuta uno de los dos apartados que hay: a) o b). Veremos que cada apartado se ejecuta en tiempo lineal, con lo que la complejidad de este paso [4] es cuadrática.

En el primero, apartado a), sólo hay que construir una lista, que ya vimos que puede hacerse en tiempo lineal, pués se trata de recorrer una curva dentro de su arreglo. Recordemos que el tiempo de acceso de una curva a su arreglo es constante, gracias al diseño de las estructuras de datos. En el segundo apartado, el b), para construir la lista hay que insertar primero el arco en el arreglo y luego borrarlo. Estas dos operaciones consumen cada una tiempo lineal.

El paso [5] es el de mezclar las cuatro listas de cada arco. Como se trata de listas ordenadas, el proceso tiene complejidad lineal para cada arco, con lo que la complejidad de este paso es cuadrática.

El paso [7] es un ciclo que se ejecuta n veces, una vez por cada circunferencia. Este ciclo tiene dos instrucciones, la primera de las cuales, inciso a), consume tiempo constante, pués se trata de concatenar dos listas ordenadas (estas listas no son otras que las listas de puntos de intersección de los dos arcos que componen la circunferencia dada). La otra instrucción, en el inciso b), tiene complejidad lineal para cada lista, como vimos en el capítulo de "Técnicas de Implementación". Podemos enunciar entonces el siguiente teorema.

Teorema 4.7.1. *Un arreglo de anillos puede construirse en tiempo $O(n^2)$.*

4.8 Simplificación de los Arreglos de Arcos

En esta sección vamos a ver que los arreglos de arcos se pueden simplificar de manera que tengan una estructura más sencilla sin que por ello se vea afectado el tiempo de construcción de los mismo utilizando un algoritmo incremental. Probaremos que es posible eliminar parte de las aristas pertenecientes a los segmentos auxiliares y seguir construyendo el arreglo en tiempo $O(n^2)$.

Recordemos que los segmentos auxiliares fueron añadidos al arreglo para garantizar la existencia de una sola componente conexa. Trajeron además como consecuencia que los arcos extendidos se comportaran como pseudorrectas. Sin embargo, para mantener el arreglo conexo durante toda su construcción basta con extender los segmentos verticales hasta el primer elemento del arreglo conque chocan (no hay que llevarlos hasta t). De esta forma se consigue que desaparezcan gran cantidad de aristas verticales que no forman parte del arreglo de arcos original. A los arreglos así modificados los llamamos arreglos simplificados.

Con esta variante, sin embargo, el conjunto de arcos extendidos no es ya un conjunto de pseudorrectas. No podemos entonces utilizar el Teorema de la Zona para obtener la complejidad de la zona de los arcos, pero sí podemos demostrar por otros medios que la complejidad de la zona de los arcos en estos arreglos es lineal.

Llamaremos igualmente arco extendido al objeto formado por un arco y sus dos segmentos verticales. La notación para el arco extendido de γ será igualmente $\overline{\gamma}$.

Construcción de los Arreglos Simplificados

Por la simetría que se da entre los cuatro conjuntos de arcos trabajaremos sólo con uno de ellos. Sea Γ_i uno de los conjuntos de arcos inferiores y $\mathcal{A}(\Gamma_i)$ el respectivo arreglo simplificado. La construcción de $\mathcal{A}(\Gamma_i)$ se hace entonces teniendo en cuenta las siguientes indicaciones:

- Se añadirá primero el segmento horizontal t y luego los arcos uno a uno.
- Cada arco se insertará con sus respectivos segmentos verticales.
- Los segmentos verticales se extenderán hacia arriba hasta chocar con algún elemento del arreglo.
- En caso de que un extremo de arco esté ya en contacto con un elemento del arreglo no se añadirá segmento vertical alguno en este extremo.
- Los arcos se van a insertar en orden decreciente de las ordenadas de sus centros.

La última indicación, la de ordenar los arcos antes de insertarlos, garantiza que el número total de segmentos verticales en el arreglo sea a lo sumo $2n$. En la figura 4.13 tenemos un ejemplo de arreglo simplificado. Obsérvese que el extremo izquierdo del arco que aparece más oscuro no tiene segmento vertical.

Observación 4.8.1. Dado que los arcos se insertan en el arreglo en orden decreciente de las ordenadas de sus centros, el arco que se inserta pueda intersectar otros arcos del arreglo pero nunca segmentos verticales.

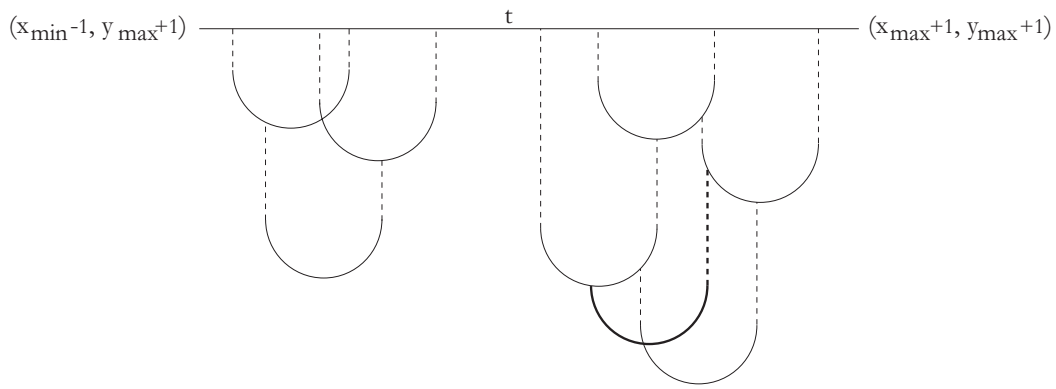


Figura 4.12: Un arreglo simplificado.

Observación 4.8.2. Las caras del arreglo que tienen sólo un punto de intersección con un arco extendido no pertenecen a la zona del mismo, como es de esperar, ya que estas caras no se visitan durante la inserción del arco. Estas caras no son otras que aquellas en donde termina la prolongación del arco.

En la siguiente figura tenemos una ilustración de esta última observación.

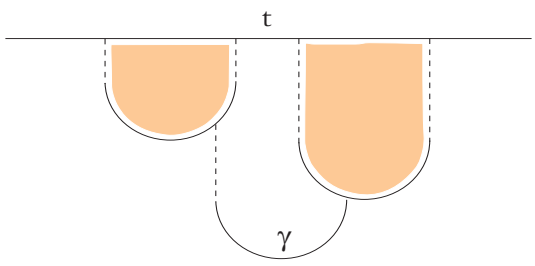


Figura 4.13: Las caras sombreadas no pertenecen a $z(\gamma)$.

4.8.1 Complejidad de $z(\gamma)$ en Arreglos Simplificados

Sea γ un arco del mismo radio y convexidad que los arcos de Γ_i. Teniendo en cuenta la definición de pseudo-convexidad vista en la sección §3.2 vemos que $\overline{\gamma}$ se comporta como un arco pseudoconvexo con respecto al conjunto de arcos extendidos de Γ_i. Podemos hablar entonces de ramas y de zona exterior de $\overline{\gamma}$ en $\mathcal{A}(\Gamma_i)$ (con la variante de que en este caso las ramas son acotadas). Probaremos que la complejidad de la zona exterior de $\overline{\gamma}$ en $\mathcal{A}(\Gamma_i)$ es lineal.

Ahora bien, como los arcos de Γ_i se intersectan dos a dos en a lo sumo un punto, cualquiera de las semizonas de $\overline{\gamma}$ puede escogerse como zona exterior. Si demostramos que cualquiera que sea la zona exterior escogida su complejidad es lineal, habremos demostrado que la zona completa de $\overline{\gamma}$ tiene complejidad lineal, y esta es la estrategia que vamos a seguir. En la figura 4.14 tenemos un ejemplo en el que aparecen las dos semizonas.

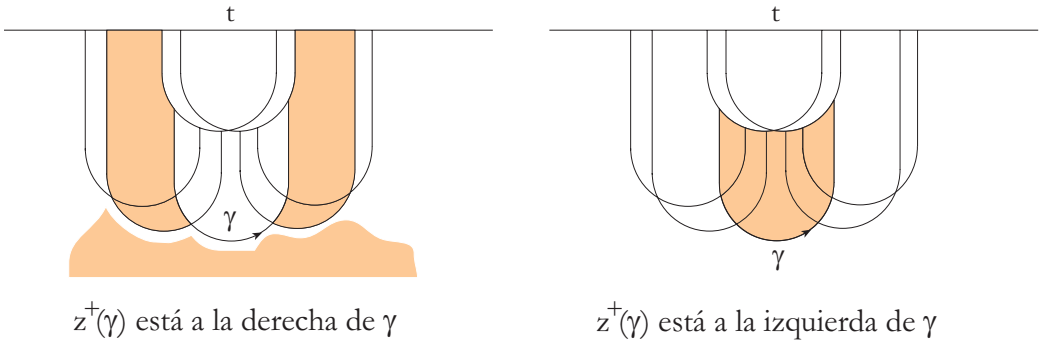


Figura 4.14: La zona exterior en cada caso.

Orientemos $\overline{\gamma}$ en un sentido cualquiera. Trabajaremos con la zona exterior que quede a la izquierda de $\overline{\gamma}$. La notación para esta zona exterior será $z^+(\gamma)$. Para probar que la complejidad de $z^+(\gamma)$ es lineal demostraremos que $z^+(\gamma)$ está formada por una cantidad lineal de aristas. La demostración para la zona exterior que queda a la derecha de $\overline{\gamma}$ es simétrica.

Nota 4.8.1. Un arco extendido $\overline{\delta}$ aporta ramas a $z^+(\gamma)$ si $\delta \cap \gamma \neq \emptyset$.

Dado que las ramas de la zona exterior de $\overline{\gamma}$ son ahora acotadas, en los arreglos simplificados podemos tener situaciones que no se habían presentado antes: caras en las que aristas consecutivas pertenecen a un mismo arco. Esto se debe a que ahora los extremos de los arcos extendidos pueden pertenecer a otro arco. En la figura 4.15 tenemos un ejemplo.

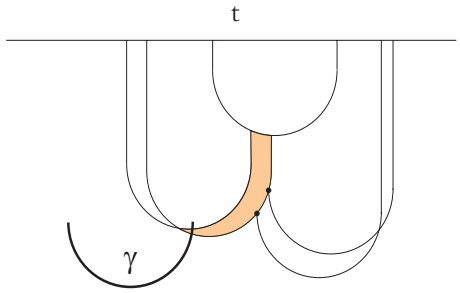


Figura 4.15: La cara sombreada tiene tres aristas consecutivas pertenecientes a un mismo arco.

Para la demostración borraremos imaginariamente, de las caras de la zona exterior, todos los extremos que hacen que aparezcan aristas consecutivas en un mismo arco. En otras palabras, las aristas consecutivas de una cara, pertenecientes a un mismo arco, serán vistas como una única arista. Con los extremos borrados, contamos el número de aristas de la zona exterior. Una vez que hayamos terminado, hacemos reaparecer los extremos. Cada extremo puede dar lugar a lo sumo a una nueva arista en la zona exterior. Al tratarse de un máximo de $2n$ extremos, la complejidad de la zona podrá aumentar en a lo sumo una cantidad lineal. Borremos estonces por un momento todos los extremos que aparezcan en caras de la zona exterior y contemos las aristas que nos quedan.

Para contar las aristas las clasificamos en izquierdas, derechas y neutrales, de modo que se facilite el conteo. La clasificación es la misma que la que vimos en la sección §3.5, pero adaptada a este caso. Sea e una arista de $z^+(\gamma)$ proveniente de $\overline{\delta}$.

Definición 4.8.1. *Decimos que e es neutral si $\delta \cap \gamma = \emptyset$.*

Sea e una arista de $z^+(\gamma)$ proveniente de $\overline{\delta}$ tal que $\delta \cap \gamma \neq \emptyset$. Sea r la rama soporte de e y $\mathcal{F}$ una de las dos posibles caras en las que e aparece.

Definición 4.8.2. *Decimos que la arista e es izquierda si $r < \mathcal{F}$ y es derecha si $\mathcal{F} < r$.*

En la figura 4.16 tenemos ejemplos de aristas izquierdas.

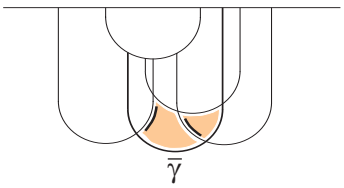


Figura 4.16: Algunas aristas izquierdas de $z^+(\gamma)$.

Lema 4.8.1. *En $z^+(\gamma)$ hay una cantidad lineal de aristas neutrales.*

Demostración. Dado que los arcos soporte de las aristas neutrales no tienen intersección con γ y que tomamos como referencia un arreglo simplificado de arcos inferiores, las aristas de tipo neutral van a encontrarse en la envolvente inferior del conjunto de arcos extendidos a los que pertenecen, como vemos en la figura 4.17.

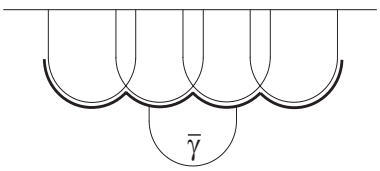


Figura 4.17: La envolvente inferior de los arcos soporte de aristas neutrales.

Por ser todos los arcos de igual radio y convexidad, cada arco puede aparecer a lo sumo una vez en esta envolvente inferior. Esto quiere decir que si en la envolvente intervienen m arcos, por lo pronto tenemos a lo sumo m aristas neutrales. Ahora tenemos que esta envolvente inferior puede ser intersectada hasta dos veces por cada uno de los restantes arcos extendidos. Cada punto de intersección dará lugar a lo sumo a una nueva arista neutral. Al tratarse de una cantidad lineal de arcos, en la envolvente inferior aparecerán a lo sumo una cantidad lineal de aristas neutrales.

□

La acotación del número de aristas izquierdas y derechas la haremos por inducción en el número de ramas. La demostración está basada en la demostración de O'Rourke en ([11], pág. 209), que está a su vez inspirada en la demostración de Edelsbrunner y otros en [5]. Utilizaremos la notación $|I|$ para referirnos al número de aristas izquierdas en la zona exterior.

Lema 4.8.2. *En $z^+(\gamma)$ hay una cantidad lineal de aristas izquierdas.*

Demostración. La hipótesis de inducción va a ser que en un arreglo $\mathcal{A}_n$ de n ramas la zona exterior tiene a lo sumo $2n$ aristas izquierdas. La hipótesis se cumple para cero ramas. Supongamos que se cumple para $n-1$. Sea r la rama de $\mathcal{A}_n$ que se intersecta con γ más a la derecha. Eliminamos r y nos queda un arreglo de $n-1$ ramas. Se cumple entonces que $|I| \leq 2(n-1)$. Llamemos $\mathcal{F}_{n-1}$ a la cara de más a la derecha en $\mathcal{A}_{n-1}$, el arreglo de $n-1$ ramas. Reinsertemos r. Veremos que r da lugar a lo sumo a dos nuevas aristas izquierdas en $z^+(\gamma)$: una en r y otra en caso de que una arista izquierda de $\mathcal{A}_{n-1}$ sea dividida por r en dos.

Cuando r reaparece en el arreglo la cara $\mathcal{F}_{n-1}$ queda dividida en dos caras que pertenecen a la zona exterior. Esto se debe a que $\mathcal{F}_{n-1}$ formaba parte de la zona exterior en $\mathcal{A}_{n-1}$ y a que la arista que la divide en dos tiene un extremo en γ. Esta arista pertenece a r y será de tipo izquierda en una de las dos caras: la que queda a la derecha de r, por definición de arista izquierda. Tenemos así una

nueva arista izquierda: la aportada por r. Ver la figura 4.18 para una ilustración.

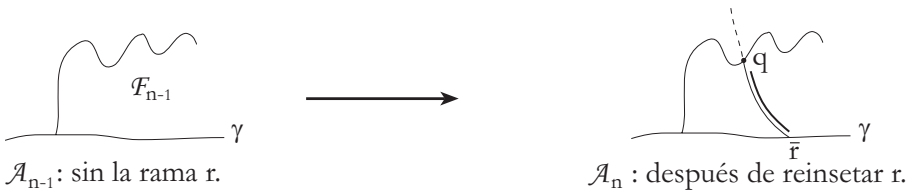


Figura 4.18: Cuando r reaparece en el arreglo.

Otra arista izquierda aparece en caso de que r intersecte la frontera exterior de $\mathcal{F}_{n-1}$. Todas las aristas de $\mathcal{F}_{n-1}$ son izquierdas o neutrales, por definición. Luego, si existe este punto de intersección q_0 entre la frontera y r, la arista en cuestión puede quedar dividida en dos aristas izquierdas que siguen perteneciendo a la zona exterior, ya que el tramo que va desde q_0 hasta la base de r tiene un extremo en γ. En la figura 4.19 tenemos una ilustración.

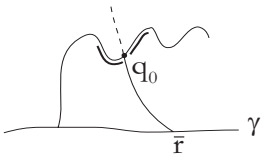


Figura 4.19: La arista izquierda es dividida en dos por q_0.

Después de pasar por q_0, r no aporta más aristas izquierdas a la zona exterior. Veremos que para este tramo de r existe una región $\mathcal{R}$ que contiene a todas las caras del arreglo adyacentes a r por su lado derecho (el lado soporte de las aristas izquierdas) y en la cual $\overline{\gamma}$ no puede entrar. Para ver quién es $\mathcal{R}$ describiremos su frontera.

De manera general, la frontera de $\mathcal{R}$ está formada por tres cadenas: dos que parten de q_0 y se extienden hasta el segmento t sin volver a intersectarse, y una tercera que pertenece a t y es la que une los extremos de las dos primeras cadenas. A las cadenas que bordean a $\mathcal{R}$ por su izquierda y por su derecha las llamaremos frontera izquierda y derecha de $\mathcal{R}$ respectivamente y las denotaremos $\mathcal{R}_1^*$ y $\mathcal{R}_2^*$.

De la figura 4.20 podemos tener una idea de cómo es $\mathcal{R}$.

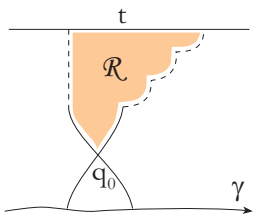


Figura 4.20: La región $\mathcal{R}$.

La frontera izquierda $\mathcal{R}_1^*$ está formada por el tramo de r comprendido entre q_0 y el extremo superior de r más el segmento vertical que une el extremo superior de r con t.

En la frontera $\mathcal{R}_2^*$ intervienen varias ramas, como vemos en la figura 4.21. Sea s_0 la rama que r intersecta en q_0. Sea q_1 el extremo superior de s_0. El primer tramo de $\mathcal{R}_2^*$ está formado por el segmento de rama q_0q_1. Sea s_1 la rama que s_0 intersecta en q_1. Sea q_2 el extremo superior de s_1. El siguiente tramo de $\mathcal{R}_2^*$ está formado por el segmento de rama q_1q_2. En q_2 repetimos el proceso y así sucesivamente hasta llegar a t.

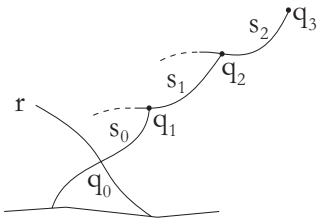


Figura 4.21: Tramos de la frontera derecha de $\mathcal{R}$.

Cuando un extremo q_i no pertenezca a una rama sino a un arco extendido s_i que no tenga intersección con γ, el tramo de s_i a añadir a $\mathcal{R}_2^*$ es el tramo q_iq_{i+1} que no tiene intersección con $\mathcal{R}_1^*$, siendo q_{i+1} uno de los extremos de s_i.

Observación 4.8.3. $\mathcal{R}_1^*$ y $\mathcal{R}_2^*$ no se intersectan después de salir de q_0.

Para verlo, basta comprobar que ningún tramo q_iq_{i+1} de $\mathcal{R}_2^*$ tiene intersección con $\mathcal{R}_1^*$, a excepción del primero: q_0q_1. Consideremos la rama extendida r^*, formada por r más un segmento vertical que une el extremo superior de r con t (r^* contiene entonces a $\mathcal{R}_1^*$). Ahora tenemos en cuenta tres hechos:

(i) El extremo q_1 del primer tramo de $\mathcal{R}_2^*$ está a la derecha de r^* (r^* es la rama de más a la derecha en γ, se intersecta con s_0 en q_0, este punto de intersección es el único entre estas ramas y q_1 aparece en s_0 después de q_0).

(ii) Si el extremo inicial de un tramo está a la derecha de r^* entonces el tramo completo está a la derecha de r^* y, por consiguiente, de $\mathcal{R}_1^*$. Esto se debe a que si el extremo q_i de q_iq_{i+1} está a la derecha de r^* es porque r^* y la rama soporte s_i del tramo se intersectan en un punto p que aparece en s_i antes que q_i, pues $r > s_i$ en γ para todo i. Ver figura 4.22.

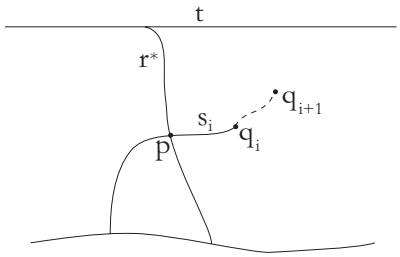


Figura 4.22: Ilustración.

Este punto p es el único punto de intersección entre r^* y s_i ($s_i \in s_i^*$, su rama extendida, y dos ramas extendidas se intersectan en un único punto). Finalmente, sabemos que q_{i+1} aparece en s_i después de q_i y, por lo tanto, después de p.

(iii) Los tramos q_iq_{i+1} que pertenecen a arcos extendidos (no tienen intersección con γ) no se intersectan con $\mathcal{R}_1^*$ por construcción. Luego, si el extremo

inicial q_i está a la derecha de r^*, el extremo final q_{i+1} queda también a la derecha.

Partiendo de (i) y teniendo en cuenta (ii) y (iii) se llega por recursividad a que después de q_0 las fronteras $\mathcal{R}_1^*$ y $\mathcal{R}_2^*$ no vuelven a intersectarse.

Una vez vistos los límites de $\mathcal{R}$ podemos probar ya que $\overline{\gamma}$ no puede entrar en $\mathcal{R}$. Primero, $\overline{\gamma}$ no tiene intersección con $\mathcal{R}_1^*$, pués $\mathcal{R}_1^*$ pertenece a la parte superior de r^*, la rama extendida de r, y ya sabemos que el único punto de intersección entre $\overline{\gamma}$ y una rama extendida es la base de ésta. Con respecto a $\mathcal{R}_2^*$, $\overline{\gamma}$ no puede intersectar los tramos provenientes de ramas por la razón que hemos visto: estos tramos están en la parte superior de las ramas. Sin embargo, $\overline{\gamma}$ sí puede intersectar los tramos provenientes de arcos extendidos (aquellos que no se intersectan con γ y, por lo tanto, no forman ramas). La intersección se produciría en este caso entre el tramo y los segmentos verticales asociados a γ. En este tipo de intersección, por construcción, no se continua con la prolongación de γ. Por la observación 4.8.2, la cara del arreglo que forma parte de $\mathcal{R}$ y contiene este punto no pertenece a $z^+(\gamma)$.

El que $\overline{\gamma}$ no pueda entrar en $\mathcal{R}$ hace que las caras de $\mathcal{R}$ adyacentes a r queden fuera de la zona exterior con lo cual, r no puede aportar nuevas aristas izquierdas a la zona exterior y que aquellas aristas izquierdas que intersecte después de q_0 serán efectivamente divididas en dos pero una de las dos partes quedará en estas caras, fuera de la zona exterior.

Queda demostrado así que la reinserción de r da lugar a lo sumo a dos nuevas aristas izquierdas. Si la acotación para el número de aristas izquierdas en $\mathcal{A}_{n-1}$ era $|I| \leq 2(n-1)$, para el arreglo $\mathcal{A}_n$ es $|I| \leq 2(n-1)+2 \leq 2n$.

□

La demostración para las aristas derechas se hace de manera simétrica. Una vez que hemos contado todas las aristas de la zona exterior provenientes de arcos extendidos sólo nos faltaría contar las aristas de la zona exterior provenientes del segmento t: por construcción, el segmento horizontal t puede ser intersectado

solamente por segmentos verticales, por lo que tendremos a lo sumo $2n+1$ aristas horizontales en todo el arreglo y, por lo tanto, en la zona exterior de $\overline{\gamma}$.

Teorema 4.8.1. *La zona exterior de $\overline{\gamma}$ en un arreglo simplificado tiene complejidad $O(n)$.*

Finalmente,

Teorema 4.8.2. *Un arreglo simplificado de arcos de circunferencia de igual radio y convexidad puede construirse en tiempo $O(n^2)$.*

Inserción de Arcos de Distinto Tipo en Arreglos Simplificados

Cuando γ no está entre los arcos del arreglo cada arco extendido puede tener hasta dos puntos de intersección con γ. Sabemos que también en este caso γ se comporta como un arco pseudoconvexo con respecto al conjunto de arcos extendidos. Podemos hablar así de la zona exterior de γ. La demostración que acabamos de ver en el apartado anterior es válida para este caso. Se prueba entonces que la zona exterior de γ tiene complejidad lineal en estos arreglos. El siguiente teorema resume este resultado.

Teorema 4.8.3. *Cuando γ no pertenece al conjunto Γ_i, puede insertarse en el arreglo simplificado $\mathcal{A}(\Gamma_i)$ en tiempo $O(n)$.*

4.9 Generalización

Consideremos el problema más general en que se tiene un conjunto Γ de n circunferencias de k radios diferentes. Utilizando el algoritmo que acabamos de explicar la construcción del arreglo de circunferencias se hace en este caso en tiempo $O(n^2\, log(k))$, como veremos a continuación. Como ya sabemos cuál es la idea del algoritmo, veremos solamente el modo en que el número de radios

diferentes interviene en la complejidad total.

- Primero, como cada una de las n circunferencias se divide en dos arcos, estamos trabajando igualmente con un conjunto de $2n$ arcos.
- Como hay k radios diferentes y los arcos se agrupan no sólo por su radio sino también por su convexidad, tenemos que en este caso se construyen $2k$ arreglos de arcos.
- La construcción de los arreglos $\mathcal{A}(\Gamma_i)$, $1 \leq i \leq 2k$, no varía en nada (sigue tratándose de arreglos de pseudorrectas). El tiempo de construcción de todos los arreglos es $O(n^2)$, como veremos posteriormente.
- La construcción de las listas sigue haciéndose en tiempo lineal para cada arco en cada arreglo. La diferencia es que ahora todo arco tendrá al final $2k$ listas asociadas.
- Mezclar las listas de un arco tiene ahora complejidad $O(n \log(k))$ por arco, ver sección § 2.4. Después de tratar los $2n$ arcos el tiempo consumido habrá sido $O(n^2 log(k))$.
- Una vez que se tienen las listas ordenadas de puntos de intersección, la construcción del arreglo de circunferencias se hace en tiempo $O(n^2)$.

Sea n_i el número de arcos en el conjunto Γ_i, $1 \leq i \leq 2k$. Sabiendo que la construcción del arreglo $\mathcal{A}(\Gamma_i)$ tiene complejidad $O(n_i^2)$ para todo i veremos que la construcción de los $2k$ arreglos tiene complejidad $O(n^2)$. Para la demostración tendremos en cuenta que

$$n_1 + n_2 + \ldots + n_{2k} = 2n,$$
$$n_i \leq n \text{ para todo } i.$$

La demostración es entonces la siguiente:

$$
\begin{aligned}
& n_1^2 + n_2^2 + \ldots n_{2k}^2 \leq \\
& \leq n \cdot n_1 + n \cdot n_2 + \ldots n \cdot n_{2k} = \\
& = n(n_1 + n_2 + \ldots n_{2k}) = \\
& = 2n^2
\end{aligned}
$$

Vistos todos los pasos, la mayor complejidad es $O(n^2\, log(k))$. Llegamos entonces a que un arreglo de n circunferencias de k radios diferentes puede construirse en tiempo $O(n^2\, log(k))$.

Antes de enunciar este resultado vamos a tener en cuenta que una vez que las circunferencias han sido divididas en dos arcos de circunferencia y que cada arco ha sido extendido con segmentos verticales (hacia arriba o hacia abajo, dependiendo de su convexidad) estamos ante un conjunto $\overline{\Gamma}$ de curvas de Jordan no acotadas que se intersectan dos a dos en a lo sumo 2 puntos. El teorema 3.7.1 nos dice que en este caso $\mathcal{A}(\overline{\Gamma})$ puede construirse en tiempo $O(n^2\, \alpha(n))$. Para pasar de $\mathcal{A}(\overline{\Gamma})$ al arreglo de circunferencias podemos ignorar los segmentos auxiliares en este arreglo o construir el arreglo de circunferencias partiendo del orden en que cada circunferencia se intersecta con las demás; orden que habremos obtenido de $\mathcal{A}(\overline{\Gamma})$. Llegamos entonces al siguiente resultado:

Teorema 4.9.1. *Un arreglo de n circunferencias de k radios diferentes puede construirse en tiempo $O(n^2 \cdot min(log(k), \alpha(n)))$.*

4.10 Extensiones del Algoritmo

En general, el algoritmo visto puede utilizarse en la construcción de arreglos de curvas que cumplan lo siguiente: el conjunto Γ de n arcos puede dividirse en k subconjuntos Γ_1, Γ_2, ..., Γ_k de manera tal que para todo $1 \leq i \leq k$:

i) Los arcos de Γ_i se extienden a un arreglo de pseudorrectas.

ii) Los arcos de Γ_i pueden insertarse en cualquier arreglo $\mathcal{A}(\Gamma_j)$, con $i \neq j$, en tiempo $O(n)$.

Teniendo como referencia el ejemplo de la sección anterior sabemos que en estos casos $\mathcal{A}(\Gamma)$ puede construirse en tiempo $O(n^2\, log(k))$.

En esta sección vamos a tratar los casos particulares en que Γ es un conjunto de n elipses o n hipérbolas de k tamaños diferentes o un conjunto de n parábolas de k amplitudes diferentes. Veremos que si todas las figuras del arreglo son isotéticas entonces la complejidad del tiempo de construcción de $\mathcal{A}(\Gamma)$ es $O(n^2\, log(k))$.

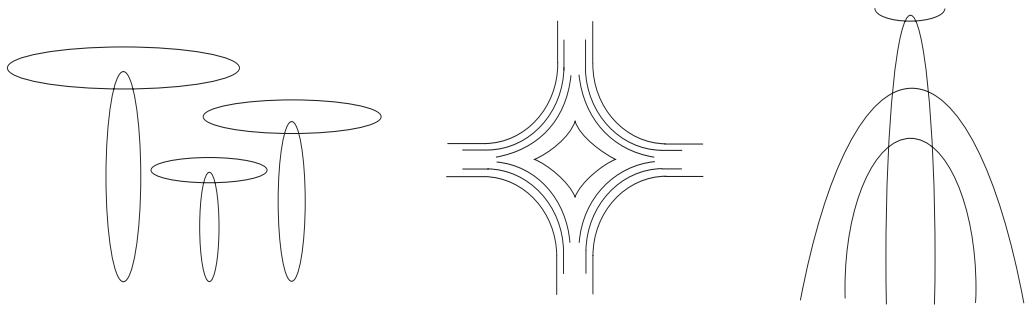


Figura 4.23: Conjuntos de Elipses, Hipérbolas y Parábolas de tamaños diferentes.

En realidad, lo que tenemos que ver es cómo dividir Γ de modo tal que los subconjuntos Γ_i obtenidos cumplan los dos requisitos anteriores. Al igual que en el caso de las circunferencias, estos arreglos podrán construirse también en tiempo $O(n^2\, \alpha(n))$ ya que, por una parte, las figuras de Γ_i van a poder extenderse a curvas de Jordan no acotadas y, por la otra, en el conjunto formado por las figuras de $\Gamma_1 \cup \Gamma_2 \cup \ldots \cup \Gamma_k$ extendidas, dos curvas van a intersectarse dos a dos en a lo sumo dos puntos, con lo que podemos aplicar el teorema 3.7.1.

4.10.1 Arreglo de Elipses

El tratamiento de las elipses se hace de manera similar al tratamiento de las circunferencias. Cada elipse perteneciente a Γ será cortada por su diámetro hori-

zontal, dando lugar a dos arcos.

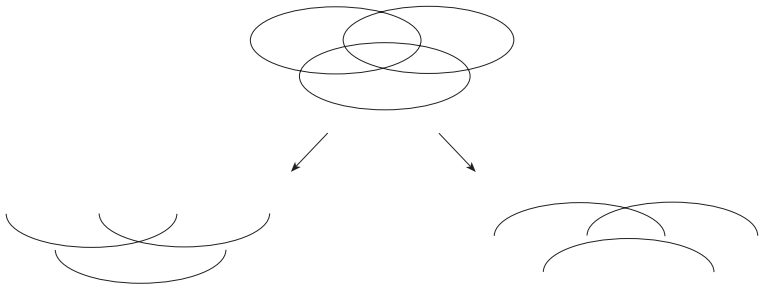


Figura 4.24: El corte en las elipses.

Como las elipses son todas isotéticas, dos de los arcos obtenidos se pueden intersectar en a lo sumo dos puntos, como puede verse en la figura 4.23. Sin embargo, dos arcos de igual tamaño y convexidad se intersectan en a lo sumo un punto, figura 4.24. Si se intersectaran en dos, los dos extremos de uno de los arcos estarían en el mismo lado respecto al otro, con lo que estaríamos ante arcos de tamaño diferente o convexidad diferente.

Cada conjunto Γ_i tendrá entonces todos los arcos que sean de igual tamaño y convexidad. n elipses de k tamaños diferentes dan lugar así a $2k$ conjuntos Γ_i. La construcción de los arreglos $\mathcal{A}(\Gamma_i)$ se hace también de manera similar a como se hizo con las circunferencias: se extienden primero los arcos con segmentos verticales en cada uno de sus extremos y se pone además un segmento horizontal t. Finalmente se comprueba que todo arco de Γ_i es siempre pseudo-convexo con respecto a los arcos de Γ_j, para todo i diferente de j. Aplicando el algoritmo visto en este capítulo llegamos a que el arreglo de elipses puede construirse en tiempo $O(n^2 \log(k))$. Considerando por otra parte el conjunto de todos los arcos extendidos y utilizando el teorema 3.7.1 llegamos a que el arreglo de elipses puede construirse en tiempo $O(n^2\,\alpha(n))$. El resultado es entonces el siguiente:

Teorema 4.10.1. *Un arreglo de n elipses de k tamaños diferentes puede construirse en tiempo $O(n^2 \cdot min(log(k), \alpha(n)))$.*

4.10.2 Arreglo de Parábolas

Dos parábolas de igual amplitud y convexidad se intersectan en a lo sumo un punto. Cuando dos parábolas se intersectan en dos puntos es porque tienen amplitudes o convexidades diferentes, como vemos en la figura 4.25.

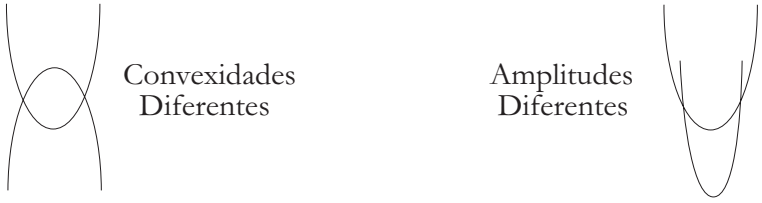


Figura 4.25: Parábolas que se intersectan en dos puntos.

La división de Γ en subconjuntos es entonces inmediata: en un mismo Γ_i estarán todas las parábolas que tengan igual amplitud y convexidad. n parábolas de k amplitudes diferentes dan lugar a k subconjuntos. Separándolas luego por convexidad se tienen $2k$ subconjuntos Γ_i. Como las parábolas son curvas no acotadas, el arreglo $\mathcal{A}(\Gamma_i)$ se construye también sin más preprocesamiento en tiempo cuadrático. Finalmente, una parábola de un conjunto cualquiera es siempre pseudoconvexa con respecto a las parábolas de otro conjunto. Gracias al algoritmo visto en esta sección y al teorema 3.7.1 llegamos al siguiente resultado.

Teorema 4.10.2. *Un arreglo de n parábolas de k amplitudes diferentes puede construirse en tiempo $O(n^2 \cdot min(log(k), \alpha(n)))$.*

4.10.3 Arreglo de Hipérbolas

Como sabemos, una hipérbola está formada por dos ramas infinitas que podemos decir tienen diferente convexidad. La división de Γ en subconjuntos se hace entonces del siguiente modo. Las hipérbolas se separan primero por tamaño, obtienendo k subconjuntos. De cada subconjunto se obtienen luego dos: uno que contiene las ramas izquierdas de las hipérbolas en cuestión y otro que contiene

las ramas derechas, para un total de $2k$ subconjuntos. En la figura 4.26 tenemos un ejemplo.

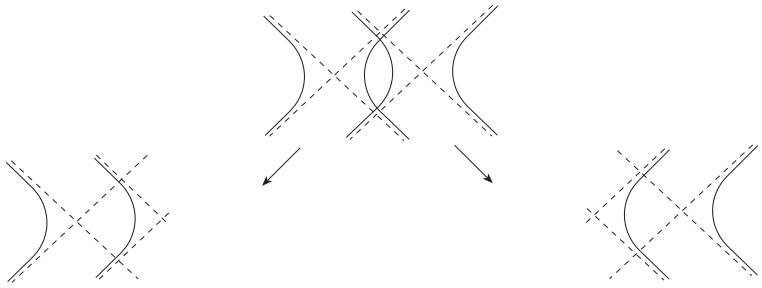


Figura 4.26: Las hipérbolas y sus ramas.

Como las ramas de las hipérbolas son curvas no acotadas que se intersectan dos a dos en a lo sumo un punto (si se intersectaran en dos, sus respectivas hipérbolas serían de tamaño diferente o se trataría de ramas de convexidad diferente, como vemos en la figura 4.27) los arreglos $\mathcal{A}(\Gamma_i)$ pueden construirse directamente en tiempo cuadrático.

Figura 4.27: Ramas de hipérbolas que pertenecen a conjuntos diferentes.

Por ota parte, es fácil ver que una rama de un conjunto Γ_i es siempre pseudo-convexa con respecto a las ramas de otro conjunto Γ_j, para todo i diferente de j.

Teorema 4.10.3. *Un arreglo de n hipérbolas de k tamaños diferentes puede construirse en tiempo $O(n^2 \cdot min(log(k), \alpha(n)))$.*

4.11 Problemas Abiertos

El problema que queda pendiente es el de hallar un algoritmo cuadrático para construir estos arreglos de n figuras geométricas del mismo tipo pero de k tamaños diferentes.

Cuando las figuras (elipses, hipérbolas y parábolas) son todas del mismo tamaño la complejidad del tiempo de construcción es $O(n^2)$, pues el conjunto inicial de curvas se divide tan sólo en dos subconjuntos. En el caso de las circunferencias existe ya un algoritmo cuadrático que permite construir el arreglo de manera directa (sin tener que dividir previamente las circunferencias en arcos). Ver el artículo de Chazelle y Lee [3].

Capítulo 5

El Problema de la Cuña

5.1 Definición del Problema

Una cuña de ángulo α es la región del plano comprendida entre dos rayos que parten de un punto común formando un ángulo α. El punto es el vértice de la cuña y los rayos, las semirrectas de la cuña. El problema que queremos resolver es el siguiente:

Problema 5. *Dado un conjunto de n puntos en el plano, un segmento $\overline{AB}$ y una cuña $\mathcal{C}$ de ángulo fijo $0 \leq \alpha \leq \pi$, hallar una posición de $\mathcal{C}$ en el segmento de modo que cubra la mayor cantidad de puntos.*

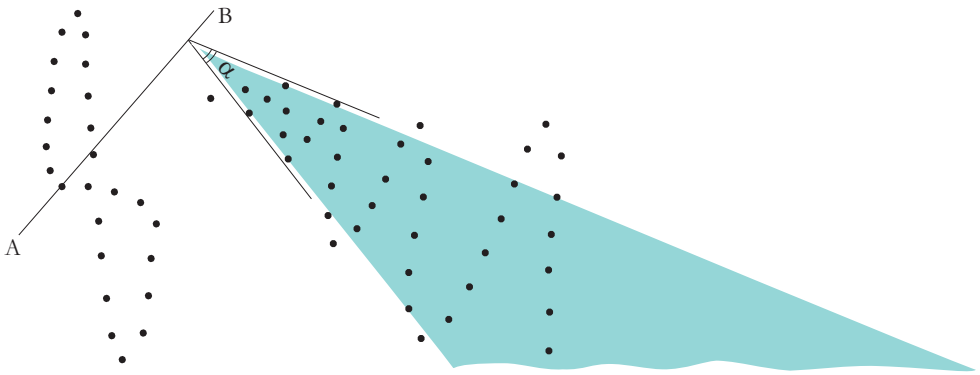


Figura 5.1: Ilustración del problema.

La posición de la cuña va a venir dada por dos valores: las coordenadas del punto donde está el vértice de la cuña y la inclinación de la cuña. Para resolver el problema supondremos, sin pérdida de generalidad, que el segmento $\overline{AB}$ está en el eje `OX`, coincidiendo A con el origen de coordenadas. La posición de la cuña es entonces (u, β), siendo u el desplazamiento del vértice con respecto al extremo A del segmento y β el ángulo que forman la semirrecta izquierda de la cuña y la semirrecta positiva del eje `OX` (mirando desde el vértice de la cuña).

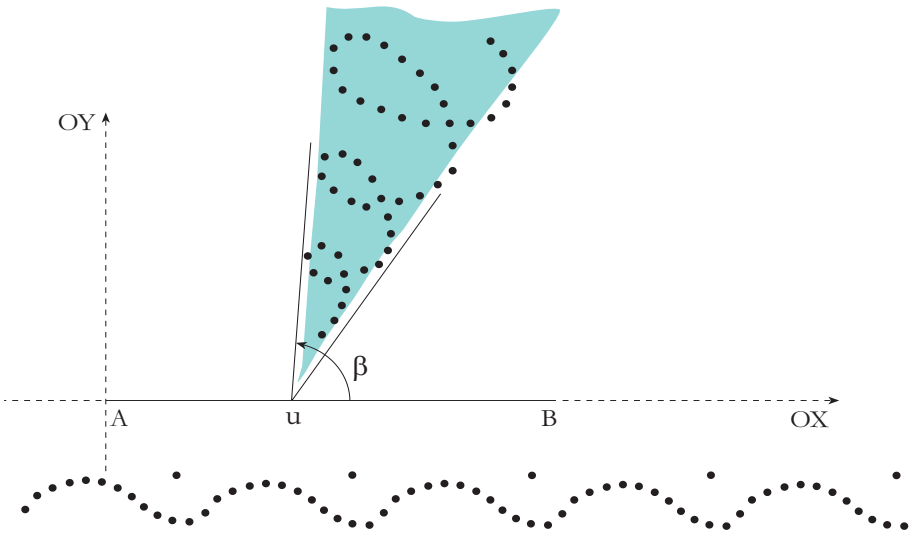


Figura 5.2: El segmento en el eje `OX`.

Sea k el número máximo de puntos que $\mathcal{C}$ puede cubrir. El conjunto solución va a estar formado por todas las posiciones en las que $\mathcal{C}$ cubre k puntos. Utilizaremos la notación $|\mathcal{C}|$ para referirnos a la cantidad de puntos cubiertos por $\mathcal{C}$.

Observación 5.1.1. El conjunto solución puede ser un conjunto infinito.

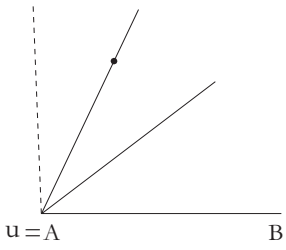
Véase que si estando $\mathcal{C}$ en una posición cualquiera donde cubre un determinado conjunto de puntos podemos rotar $\mathcal{C}$ o trasladarla a lo largo de $\overline{AB}$ sin perder ninguno de los puntos cubiertos, estaremos visitando infinitas posiciones donde $\mathcal{C}$ cubre la misma cantidad de puntos.

Observación 5.1.2. El conjunto solución infinito está representado por un conjunto solución finito.

En la siguiente sección vamos a ver que existen tres posiciones características para la cuña de modo que cualquier posición para $\mathcal{C}$ puede transformarse en una de estas tres posiciones características. La solución se busca entonces dentro del conjunto de posiciones caraterísticas, que será un conjunto finito, como veremos a continuación. Como en nuestro problema lo que se quiere es hallar una posición donde $\mathcal{C}$ cubra la mayor cantidad de puntos, no tenemos que dar las infinitas soluciones sino que bastará con algunas de ellas. Dependiendo del algoritmo utilizado, sin embargo, el conjunto solución que se obtiene es unas veces finito y otras infinito.

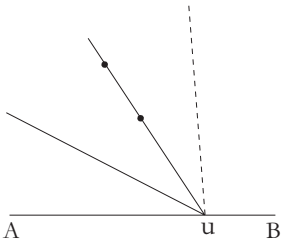
5.2 Posiciones Características

Sea (u, β) una posición cualquiera de $\mathcal{C}$. Hagamos rotar $\mathcal{C}$ hasta que una de las semirrectas choque con un punto. Manteniendo ahora este punto en la semirrecta, traslademos $\mathcal{C}$ a lo largo de $\overline{AB}$ hasta que choque con otro punto o su vértice llegue a uno de los extremos del segmento. Si en ese momento miramos la cuña, ésta tendrá una de las tres configuraciones siguientes.



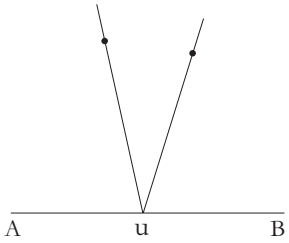
Configuración (i):
El vértice está en A o B y
una de las semirrectas pasa por al menos un punto.

Como cada posición está generada aquí por un punto el número de posiciones donde $\mathcal{C}$ tiene configuración (i) va a ser una cantidad lineal. En ocasiones nos referiremos a la configuración (i) como la configuración lineal.



Configuración (ii):
El vértice pertenece al interior del segmento y una de las semirrectas pasa por al menos dos puntos.

Tanto en esta configuración (ii) como en la siguiente cada posición viene determinada por dos puntos. La cantidad total de posiciones donde $\mathcal{C}$ tiene configuración (ii) o (iii) va a ser, por lo tanto, una cantidad cuadrática: $\binom{n}{2}$. En ocasiones nos referiremos a estas configuraciones como las configuraciones cuadráticas.



Configuración (iii):
El vértice pertenece al interior del segmento y cada una de las semirrectas pasa por al menos un punto.

La configuración en la que $\mathcal{C}$ termina va a depender del orden en que se hagan las transformaciones (rotación y traslación) y de la orientación que se escoja para las mismas.

Si denotamos por $(u, \beta)^*$ al conjunto de puntos que $\mathcal{C}$ cubre en (u, β) tenemos entonces que para toda posición (u, β) de $\mathcal{C}$ en el segmento existe una posición (u', β') que podemos llamar "equivalente" en la que $(u, \beta)^* \subseteq (u', \beta')^*$ y $\mathcal{C}$ tiene una de las tres configuraciones anteriores. Cualquier posición que forme parte de la solución va a tener también, por lo tanto, su posición equivalente. De ahora en adelante nos dedicaremos a este conjunto de posiciones, aquellas donde $\mathcal{C}$ tiene configuración (i), (ii) o (iii).

Para hallar la solución del problema vamos a hallar la solución para cada una de las configuraciones por separado. A estas soluciones las llamaremos *soluciones parciales*. El esquema general del algoritmo es entonces bien sencillo.

Algoritmo General

[1] Hallar la solución parcial para cada una de las configuraciones.

[2] Hacer k_1, k_2 y k_3 igual al número máximo de puntos cubiertos cuando $\mathcal{C}$ tiene configuración (i), (ii) y (iii) respectivamente.

[3] Hacer $k = \text{máximo}\{k_1, k_2, k_3\}$.

[4] Devolver la solución parcial de las configuraciones donde se cubrieron k puntos.

5.3 Solución para la Configuración (i)

En esta configuración el vértice de la cuña está fijo en uno de los extremos del segmento. El conjunto de las posiciones candidatas a solución puede dividirse entonces en dos subconjuntos: $\{(A, \beta_1), (A, \beta_2), \ldots, (A, \beta_{2n})\}$ y $\{(B, \beta_1), (B, \beta_2), \ldots, (B, \beta_{2n})\}$. Por la simetría que se da entre los dos subconjuntos trabajaremos con sólo uno de ellos: aquel en el que $u = A$, por ejemplo. Véase que el tamaño de los subconjuntos es $2n$. Esto se debe a que cada punto da lugar a dos posiciones para $\mathcal{C}$, como vemos en la figura 5.3: una en la que la semirrecta izquierda pasa por el punto y otra en la que es la semirrecta derecha la que pasa por el punto. La proposición que tenemos a continuación, sin embargo, nos permite dedicarnos a sólo uno de los dos casos.

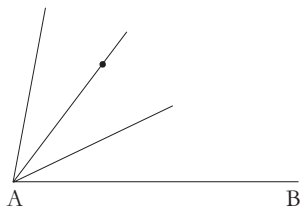


Figura 5.3: Las dos posiciones a que da lugar p.

Proposición 5.3.1. *Toda posición (A, β) de $\mathcal{C}$ en la que la semirrecta derecha pasa por al menos un punto tiene una posición equivalente (A, β') en la que la*

semirrecta izquierda pasa por al menos un punto y $(A, \beta)^* \subseteq (A, \beta')^*$.

Demostración-. Si en la posición de partida $\mathcal{C}$ tiene también un punto en su semirrecta izquierda, hemos terminado. Supongamos entonces que la semirrecta izquierda de $\mathcal{C}$ está libre de puntos.

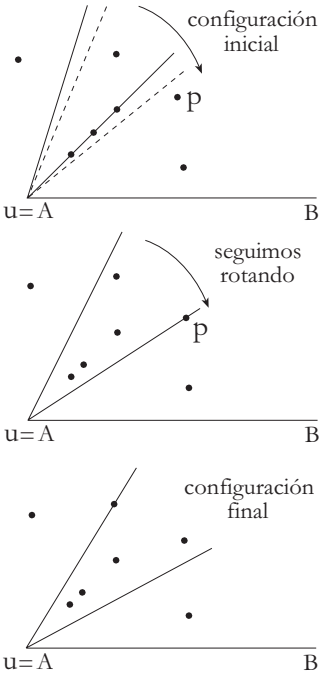


Figura 5.4: Transformación.

Hagamos rotar $\mathcal{C}$ en sentido horario alrededor de su vértice hasta que choque con un punto. Pueden darse dos casos; o bien la semirrecta izquierda choca con un punto, y hemos terminado, o bien la semirrecta derecha encuentra un nuevo punto, y entonces tenemos que seguir rotando, aumentando así el número de puntos que quedan en el interior de $\mathcal{C}$.

Rotaremos hasta tener al menos un punto sobre la semirrecta izquierda y esto siempre sucederá, ya que la semirrecta izquierda chocará en última instancia con alguno de los puntos que estaban al inicio sobre la semirrecta derecha.

□

Teniendo en cuenta esta proposición trabajaremos sólo con uno de los dos casos: aquel en el que hay al menos un punto en la semirrecta izquierda. Hemos terminado así con un subconjunto $\{(A, \beta_1), (A, \beta_2), \ldots, (A, \beta_n)\}$ de n posiciones candidatas a solución en el que todas tienen el mismo vértice.

Para buscar la solución dentro del subconjunto anterior el primer paso del algoritmo será generar las posiciones (A, β_i). Otro de los pasos va a ser determinar el número de puntos que se cubren en cada posición. Ahora bien, visitar las n posiciones y en cada una repasar los n puntos para ver cuál está cubierto

por la cuña en ese momento y cuál no nos lleva a un algoritmo de complejidad cuadrática. Sin embargo, si entre los dos pasos anteriores intercalamos uno que ordene las posiciones según el valor de β_i, la complejidad del algoritmo baja de $O(n^2)$ hasta $O(n\ \log(n))$. Esta misma idea la veremos luego en el tratamiento de las configuraciones cuadráticas, a pesar de que con estas configuraciones el vértice de $\mathcal{C}$ puede moverse a lo largo del segmento. En ese caso la complejidad bajará de $O(n^3)$ a $O(n^2\ \log(n))$.

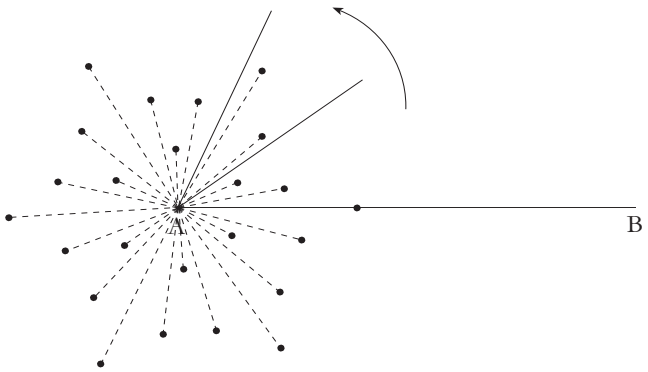


Figura 5.5: Los puntos ordenados alrededor de A.

Ordenar las posiciones según el valor de β_i es lo mismo que ordenar los puntos angularmente alrededor de A (recordemos que β_i es el ángulo de inclinación de la semirrecta izquierda y que en las configuraciones que estamos tratando los puntos generadores de las posiciones están en la semirrecta izquierda). Gráficamente, una vez que tenemos los puntos ordenados alrededor de A hacemos que $\mathcal{C}$ pase por todas las posiciones de manera que al hacerlo esté rotando alrededor de A en el mismo sentido en el que fueron ordenados los puntos. Ver la figura 5.5. De este modo, $\mathcal{C}$ barre todo el plano mientras los puntos entran y salen de $\mathcal{C}$. Finalmente, para saber cuántos puntos se cubren en cada posición sólo es necesario mirar los n puntos una sola vez: cuando $\mathcal{C}$ está en la posición de partida. Después de este momento, a medida que la cuña avanza de posición en posición, sólo hay que llevar la cuenta de los puntos que van entrando y saliendo de $\mathcal{C}$. De manera general, los pasos del algoritmo son los que tenemos a continuación. Para cada paso veremos ahora los detalles.

Algoritmo Vértice Fijo

[1] Generar todas las posiciones de $\mathcal{C}$ en las que el vértice está fijo en A y la semirrecta izquierda pasa por un punto.

[2] Ordenar las posiciones en sentido anti-horario y considerar (A,β_1) como la primera.

[3] Poner $\mathcal{C}$ en (A,β_1) e inicializar la lista de soluciones $\mathcal{L}$ con este valor.

[4] Hacer $k = |\mathcal{C}|$ y $k_{\max} = |\mathcal{C}|$.

[5] Mientras la posición de $\mathcal{C}$ no sea la última:

a) Pasar a la siguiente posición (A,β):

b) Hacer $k = |\mathcal{C}|$.

c) Si $k > k_{\max}$ hacer $k_{\max} = k$ y vaciar $\mathcal{L}$.

d) Si $k = k_{\max}$ añadir (A,β) a $\mathcal{L}$.

[6] Devolver $\mathcal{L}$ y $k_{\max}$.

El paso [1] ya lo hemos visto y el paso [2] es de ordenación. Solamente recordaremos que ordenar posiciones es lo mismo que ordenar puntos y que este orden es circular. En el paso [3], poner $\mathcal{C}$ en (A, β_1) significa poner la semirrecta izquierda pasando por el primer punto respecto al orden angular. Una vez fija la semirrecta izquierda, la posición de la semirrecta derecha queda también establecida, por ser la cuña de ángulo fijo, y sólo tenemos que buscarla. Para determinar entre qué puntos se encuentra visitamos la lista ordenada de puntos (moviéndonos ahora en sentido horario), avanzando mientras el ángulo entre las semirrectas sea menor o igual que α. Por cada punto que vaya pasando al interior de la cuña incrementamos en uno $|\mathcal{C}|$, que tendrá inicialmente el valor cero. Una vez que terminamos de situar las semirrectas en la posición de partida inicializamos las variables k y k_{max}. A partir de este momento comienza la rotación de $\mathcal{C}$, paso [5].

Durante la rotación las dos semirrectas van a moverse en un mismo sentido: el sentido en que fueron ordenados los puntos. Cada una de las semirrectas se moverá por separado y partiendo siempre de la posición anterior. De las dos semirrectas, la izquierda será la primera en avanzar y lo hará de punto en punto. Una vez fijada su posición, la semirrecta derecha rotará mientras el ángulo de la cuña sea mayor que α. Por cada nueva posición de la semirrecta izquierda aumentará en uno el valor de $|\mathcal{C}|$ y por cada punto encontrado por la semirrecta derecha el valor de $|\mathcal{C}|$ disminuirá en uno. La rotación termina cuando $\mathcal{C}$ llega a la última de las posiciones.

Análisis de la Complejidad

Para ordenar los puntos utilizamos el algoritmo MergeSort o cualquier algoritmo que trabaje en tiempo $O(n\ \log(n))$. La complejidad en cuanto al tiempo del procedimiento que halla la solución para la configuración (i) es entonces $O(n\ \log(n))$. Véase que después de la ordenación de los n puntos en [2], el resto de los pasos tiene complejidad a lo sumo lineal: la semirrecta izquierda visita cada punto solamente una vez y la semirrecta derecha, a lo sumo dos veces. La ordenación de los puntos es el paso que determina la complejidad total.

Lema 5.3.1. *La posición con configuración (i) donde $\mathcal{C}$ cubre la mayor cantidad de puntos puede hallarse en tiempo $O(n\,log(n))$.*

Demostración. El resultado se deduce de la proposición 5.3.1 y del algoritmo Vértice Fijo, teniendo en cuenta que la solución con la cuña en el extremo B del segmento se calcula del mismo modo que en el extremo A.

□

Observación 5.3.1. Se puede modificar el algoritmo para obtener las infinitas soluciones, en vez de un conjunto discreto, sin que se altere la complejidad.

Las infinitas soluciones estarán contenidas dentro de intervalos. Para trabajar con esta alternativa bastará con añadir variables adicionales que vayan llevando la cuenta de los extremos de los sucesivos intervalos.

5.4 Solución para las Configuraciones (ii) y (iii)

Estas dos configuraciones las vamos a analizar dentro de una misma sección porque, como veremos, pueden ser consideradas equivalentes. Las propiedades 5.4.1 y 5.4.2 que tenemos más adelante nos muestran que partiendo de una cualquiera de las dos configuraciones se puede llegar siempre a la otra o a la configuración (i). Si tenemos en cuenta que la configuración (i) se analiza aparte, la llamada equivalencia significa que no hay que considerar estas dos configuraciones a la vez cuando se está buscando la solución parcial en el conjunto de posiciones con configuración cuadrática.

Veremos tres procedimientos diferentes: uno de complejidad $O(n^2 \log(n))$, otro de complejidad $(n^2\,\alpha(n))$ y un tercero de complejidad $O(n^2)$. Puesto que la solución parcial para la configuración (i) se halla en tiempo menor, el procedimiento que se utilice para hallar la solución parcial de las configuraciones cuadráticas es el que dará la complejidad del algoritmo general. Antes de seguir vamos a ver que al manejar las configuraciones cuadráticas no tenemos que trabajar a la vez con todos los puntos del conjunto de entrada, lo que simplificará el tratamiento del problema.

Simplificación del Problema

Consideremos la recta que contiene al segmento $\overline{AB}$. Esta recta divide al plano en dos semiplanos. De los puntos que pertenezcan a un mismo semiplano diremos que están hacia un mismo lado de $\overline{AB}$.

Observación 5.4.1. Para hallar la solución parcial del conjunto de posiciones con configuración cuadrática podemos suponer que los puntos con los que trabajamos están todos hacia un mismo lado de $\overline{AB}$.

Si tenemos una solución en la que $\mathcal{C}$ cubre puntos a ambos lados de $\overline{AB}$ siempre podemos trasladar $\mathcal{C}$ hasta uno de los extremos del segmento sin perder ninguno de los puntos cubiertos. La cuña termina así en una posición con configuración (i), que hemos analizado aparte. Ver figura 5.6.

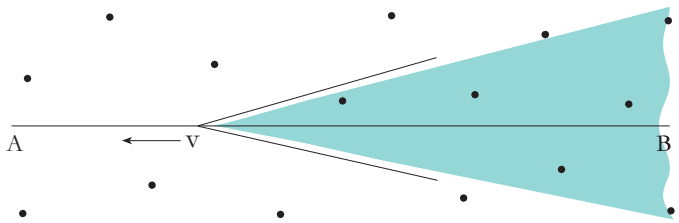


Figura 5.6: Ilustración.

En los tres procedimientos que vamos a ver hemos supuesto, sin pérdida de generalidad, que los puntos se encuentran por encima de $\overline{AB}$. Finalmente, para dar la solución parcial del conjunto de posiciones con configuración cuadrática el procedimiento que se emplee tendrá que ser aplicado dos veces: una en cada lado de $\overline{AB}$. En caso de existir puntos que pertenezcan al segmento los incluimos en los dos análisis.

Equivalencia entre las Configuraciones (ii) y (iii)

Propiedad 5.4.1. *Para toda posición (u, β) en la que $\mathcal{C}$ tiene configuración (iii) existe una posición (u', β') en la que $(u, \beta)^* \subseteq (u', \beta')^*$ y $\mathcal{C}$ tiene configuración (ii) o configuración (i).*

Demostración. Sean p_i y p_j los puntos que están en las semirrectas de $\mathcal{C}$. Supongamos, sin pérdida de generalidad, que p_i pertenece a la semirrecta izquierda y p_j a la derecha. Hallemos la circunferencia que contiene al arco capaz de ángulo α que pasa por p_i y p_j. Los puntos de intersección entre esta circunferencia y el segmento, que tienen que existir, no son más que los vértices de aquellas posiciones que tienen a p_i y p_j en las semirrectas izquierda y derecha respectivamente. Llamemos u_1 y u_2 a estos puntos de intersección, siendo u_1 menor que u_2.

Supongamos, sin pérdida de generalidad, que $\mathcal{C}$ está en u_1.

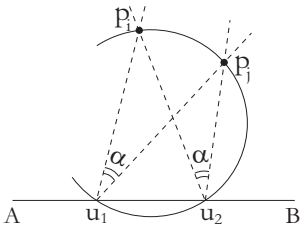


Figura 5.7: Arco Capaz de ángulo α que pasa por p_i y p_j.

Manteniendo a p_i y p_j en las semirrectas, traslademos el vértice de $\mathcal{C}$ hacia la izquierda hasta que una de las semirrectas choque con un punto o el vértice llegue al extremo A del segmento. Ver figura 5.8.

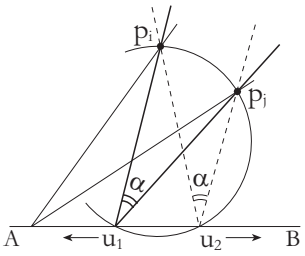


Figura 5.8: Traslación partiendo de la configuración (iii).

Si $\mathcal{C}$ está inicialmente en u_2, la traslación se haría hacia B. Supongamos que antes de llegar al extremo A la semirrecta izquierda choca con un punto. Por estar u_1 fuera de la circunferencia en ese momento, el ángulo entre las semirrectas se habrá hecho menor que α. Podemos entonces hacer rotar la semirrecta derecha hasta que el ángulo vuelva a ser α. De esta manera, sin perder ninguno de los puntos cubiertos inicialmente, hemos llegado a una posición en la que $\mathcal{C}$ tiene configuración (ii). Ver la figura 5.9 en la página siguiente.

En caso de no encontrar ningún punto durante la traslación $\mathcal{C}$ termina con configuración (i). El mismo resultado se obtiene si al trasladar $\mathcal{C}$ hacia A la semirrecta que choca con un punto es la derecha.

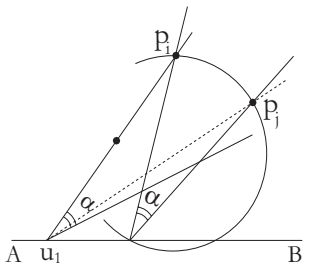


Figura 5.9: Configuración (ii).

□

Esta propiedad nos permite olvidarnos de las posiciones de configuración (iii) y considerar solamente las posiciones donde $\mathcal{C}$ tiene configuración (ii). Con la siguiente propiedad los papeles de las configuraciones (ii) y (iii) se invierten.

Propiedad 5.4.2. *Para toda posición (u, β) en la que $\mathcal{C}$ tiene configuración (ii) existe una posición (u', β') en la que $(u, \beta)^* \subseteq (u', \beta')^*$ y $\mathcal{C}$ tiene configuración (iii) o configuración (i).*

Demostración-. Supongamos, sin pérdida de generalidad, que p_i y p_j se hallan en la semirrecta izquierda de $\mathcal{C}$ y que p_j tiene menor ordenada que p_i.

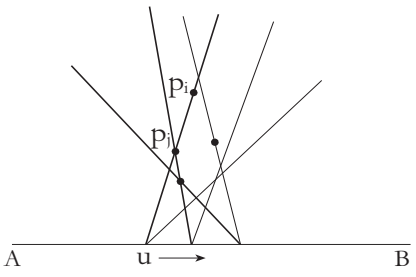


Figura 5.10: Traslación.

Manteniendo a p_j en la semirrecta izquierda trasladamos $\mathcal{C}$ hacia la derecha. Como resultado, la semirrecta izquierda va a girar alrededor de p_j en sentido antihorario. Cada vez que esta semirrecta choque con un punto, actualizamos el punto de menor ordenada y continuamos la traslación apoyándonos en él. Paramos cuando la semirrecta derecha choque con un punto o lleguemos al extremo del segmento.

Del mismo modo podíamos haber hecho la traslación hacia la izquierda, actualizando como punto de apoyo para la rotación el punto de mayor ordenada y haciendo rotar la semirrecta izquierda en sentido horario. □

5.4.1 Algoritmo de las Particiones

La idea básica de este algoritmo es similar a la que utilizamos para buscar la solución con configuración (i): generar todas las posiciones y visitarlas luego de una en una, contando en cada posición el número de puntos que se cubren. Sin hacer ningún otro procesamiento la complejidad del algoritmo sería cúbica: $\binom{n}{2}$ posiciones que operan con n puntos cada una; pues en este caso cada posición está generada por dos puntos.

Para bajar la complejidad hasta $O(n^2 \log(n))$ utilizamos también la idea que usamos para la configuración (i): tener el orden en que los puntos aparecen alrededor del vértice de la cuña. En este caso, sin embargo, el vértice puede moverse a lo largo del segmento, por lo que no es buena idea ordenar los puntos para cada nueva posición si queremos que la complejidad baje de $O(n^3)$. En su lugar, utilizaremos la estrategia de ordenar los puntos una primera vez y actualizar este orden en lo sucesivo. El primer paso de este algoritmo será entonces ordenar los puntos angularmente alrededor de A (la ordenación se hará en sentido antihorario). La propiedad que tenemos a continuación nos va a permitir actualizar el orden de los puntos en tiempo constante para cada posición. A fin de evitar que las situaciones degeneradas oscurezcan la idea central del algoritmo supondremos por el momento que no hay dos posiciones que coincidan en cuanto al valor del vértice. De esta suposición se deduce que no encontraremos más de dos puntos alineados ni posiciones donde la cuña tenga más de dos puntos en su frontera.

Propiedad 5.4.3. *El orden angular de los puntos sólo cambia en aquellos tramos del segmento que contienen posiciones alineadas con al menos dos puntos.*

Sean p y q dos puntos cualesquiera tales que la recta r que los contiene se intersecta con el segmento en a. Supongamos que p es el punto de mayor ordenada

(q puede pertenecer al segmento o no). Se tiene entonces que para toda $x < a$ el valor angular de p es mayor que el de q y que para toda $x > a$ ocurre a la inversa y el valor angular de p es menor. En la figura 5.11 tenemos una ilustración. La notación $p > q$ significa que el valor angular de p es mayor que el de q. De manera similar para $p < q$.

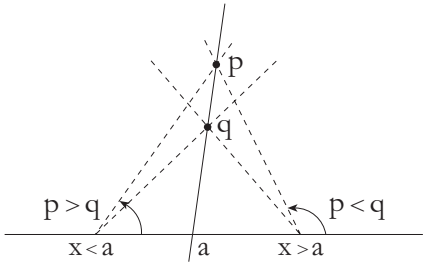


Figura 5.11: Inversión del orden angular de los puntos al pasar por a.

Para mantener el orden de los puntos, una vez que fueron ordenados angularmente con respecto a A, es preciso que las posiciones donde el orden se altera sean visitadas también de manera ordenada. En el ejemplo de la figura 5.12 el orden de los puntos con respecto a x e y es $prqs$ y $srqp$ respectivamente. a, b, c, d y e son todas las posiciones de $\overline{AB}$ alineadas con al menos dos puntos.

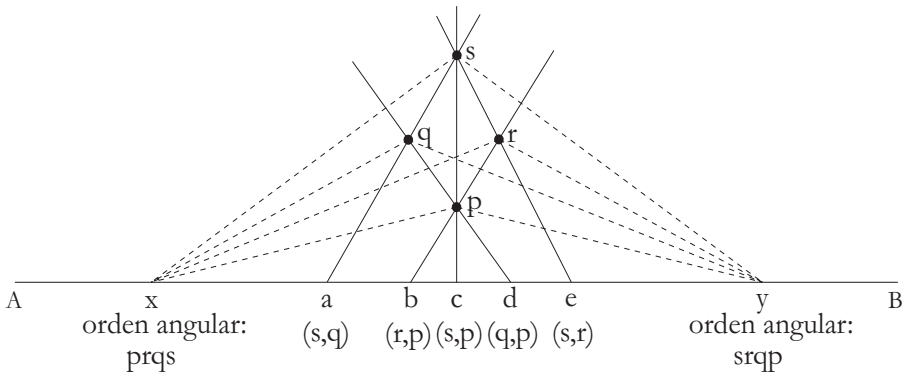


Figura 5.12: Ejemplo.

El orden angular de los puntos con respecto a b no va a ser el mismo que con respecto a x ya que el tramo xb contiene a la posición a que, al estar alineada con los puntos s y q provoca que el orden entre éstos se invierta al pasar de las posiciones a la izquierda de a a las posiciones a la derecha.

Otro de los pasos del algoritmo va a ser entonces ordenar estas posiciones a lo largo de $\overline{AB}$. Y quiénes son estas posiciones. Estas posiciones no son otras que aquellas donde $\mathcal{C}$ tiene configuración (ii). Recuérdese que la configuración (ii) es la única que tiene al menos dos puntos alineados según una recta que se intersecta con $\overline{AB}$. Llegamos así a que, visitando las posiciones de configuración (ii) de manera ordenada, la actualización del orden de los puntos se hace en tiempo constante para cada posición.

Una vez que está garantizado el orden angular con respecto a cada tramo del segmento el siguiente objetivo es contar el número de puntos que se cubren en cada una de las diferentes posiciones de $\mathcal{C}$. Este paso se hará también en tiempo constante gracias a la siguiente propiedad.

Propiedad 5.4.4. *Si tenemos un conjunto de puntos ordenados podemos saber en tiempo constante la cantidad de puntos que hay entre dos de ellos.*

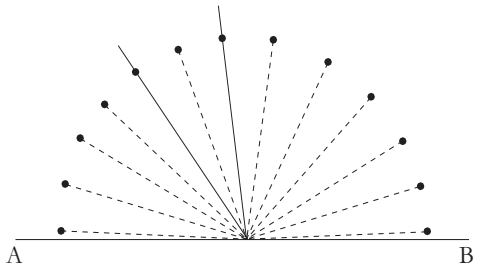


Figura 5.13: Contando el número de puntos cubiertos en vértices de configuración (iii).

Cuando $\mathcal{C}$ tiene configuración (iii) cada una de sus semirrectas pasa por al menos un punto. Esto quiere decir que teniendo el orden en que los puntos aparecen

alrededor del vértice y teniendo la localización de las semirrectas dentro de ese orden podemos saber en tiempo constante el número de puntos que $\mathcal{C}$ cubre.

Localizar las semirrectas dentro del conjunto de puntos equivale a localizar los puntos generadores dentro del orden angular de puntos ya que, en las configuraciones de tipo (iii), cada punto pertenece a una semirrecta. Finalmente, gracias a la propiedad de equivalencia entre las configuraciones (ii) y (iii), el cálculo del número de puntos cubiertos se hará sólo en las posiciones de configuración (iii).

La estrategia de este primer algoritmo consiste entonces en recorrer el segmento de A a B y, para cada posición encontrada, hacer la operación que corresponda según sea la configuración de $\mathcal{C}$. El esquema general del algoritmo es el siguiente:

Algoritmo de las Particiones

[1] Ordenar los puntos angularmente alrededor de A.

[2] Generar todas las posiciones donde $\mathcal{C}$ tiene configuración (ii) y (iii).

[3] Ordenar las posiciones en sentido creciente de sus abscisas.

[4] Inicializar k y $k_{\max}$ a cero y $\mathcal{L}$ a lista vacía.

[5] Visitar las posiciones (ν,β) de manera ordenada y en cada una:

a) Si $\mathcal{C}$ tiene configuración (ii):

I) actualizar el orden angular de los puntos.

b) Si tiene configuración (iii):

I) hacer k igual a la cantidad de puntos cubiertos.

II) si $k > k_{\max}$ hacer $k_{\max} = k$ y vaciar $\mathcal{L}$.

III) si $k = k_{\max}$ añadir (ν,β) a $\mathcal{L}$.

[6] Devolver $\mathcal{L}$ y $k_{\max}$.

El primer paso del algoritmo, en el que ordenamos los puntos alrededor de A, es el paso que determina el nombre de los puntos. Esto quiere decir que después de realizada esta operación el punto que quedó en la posición k será, hasta el final del algoritmo, el punto p_k.

El orden relativo de los puntos se almacenará en un vector P cuyo índice nos dirá el punto con el que estamos tratando y, el contenido, la posición que ocupa el punto dentro del orden angular. De esta manera, si en un determinado momento p_i es el k-ésimo de los puntos, la información almacenada en el vector será de la forma $P[i] = k$. En la figura 5.14 tenemos un ejemplo.

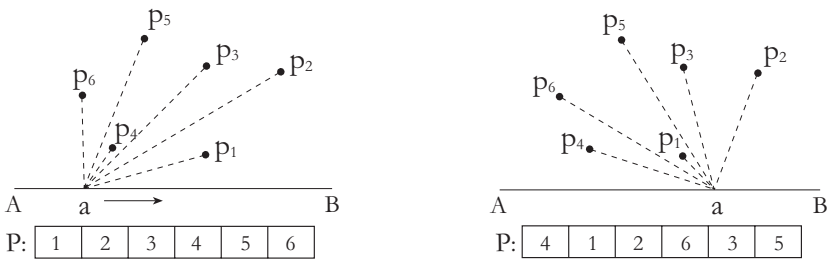


Figura 5.14: La entrada $P[i]$ guarda la posición de p_i.

En la figura de la izquierda los puntos aparecen con el orden inicial, de ahí que cada entrada del vector coincida con la posición del punto dentro del orden. En la figura de la derecha el orden de los puntos ha cambiado. El punto p_1, por ejemplo, ha pasado a ser el cuarto, de ahí que la información almacenada en P sea $P[1] = 4$.

Para generar las posiciones en el paso [2] hay que calcular primero todas las combinaciones de dos puntos y luego, para cada par de puntos, hallar todas las posiciones en las que $\mathcal{C}$ tiene configuración (ii) o (iii). Todo par de puntos dará lugar, a lo sumo, a seis posiciones diferentes: dos de configuración (ii) y cuatro de configuración (iii), como vemos en la figura 5.15.

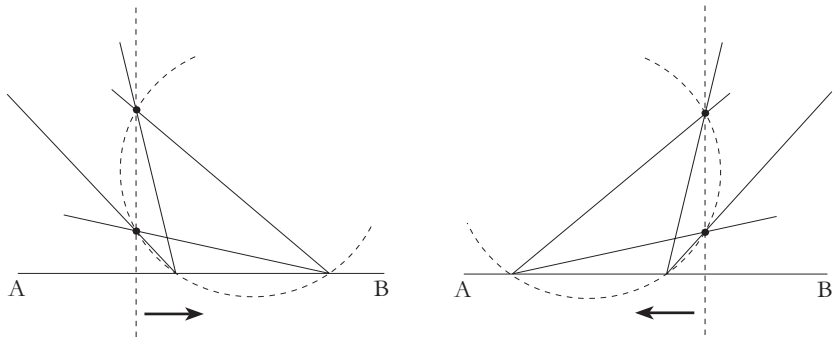


Figura 5.15: Posiciones generadas por este par de puntos.

- Las dos posiciones de configuración (ii) tendrán igual vértice pero diferente valor de β. En una de las posiciones los dos puntos generadores pertenecen a la semirrecta izquierda y en la otra, a la semirrecta derecha. Estas posiciones están en los puntos de intersección del segmento con la recta que pasa por el par de puntos.

- Las cuatro posiciones de configuración (iii) están en los puntos de intersección del segmento con los dos arcos capaces de ángulo α que pasan por los dos puntos. Cada arco capaz se obtiene a partir del otro mediante una reflexión con respecto a la recta que contiene al par de puntos.

Las posiciones se almacenarán como cuaternas $((u, \beta), c, s, (p, q))$, de modo que podamos tratar los casos degenerados. La componente c se refiere al tipo de configuración. En caso de que se trate de la configuración (ii) s indica en qué semirrecta se encuentran los puntos, si en la izquierda o en la derecha y (p, q) son los puntos generadores de la posición. En caso de que sea una posición de configuración (ii), el punto p es el de menor ordenada.

En el paso [5], cuando se trate de una posición de configuración (ii) con puntos asociados p_i y p_j, actualizar el orden angular se traduce en intercambiar el contenido de las entradas i y j de P. Dentro de este mismo paso, cuando se trate de la configuración (iii), si los puntos que la definen son p_i y p_j, la cantidad de puntos cubiertos se obtiene haciendo $|P[i] - P[j]| + 1$.

Análisis de la Complejidad

La operación más costosa del Algoritmo de las Particiones es la del paso [3]: ordenar el conjunto de posiciones. Al tratarse de un conjunto de tamaño cuadrático, la complejidad de este paso va a ser $O(n^2 \log(n))$.

El resto de las operaciones se realiza cada una en tiempo a lo sumo cuadrático. En el paso [1] trabajamos con un conjunto de tamaño lineal y el tiempo a consumir será $O(n \log(n))$. El paso [2] consume tiempo cuadrático y el paso [5] se ejecuta para un número cuadrático de elementos, cada uno de los cuales se procesa en tiempo constante.

Lema 5.4.1. *El Algoritmo de las Particiones tiene complejidad* $O(n^2 \log(n))$.

Casos Degenerados

Las degeneraciones se dan cuando $m > 1$ posiciones de $\mathcal{C}$ comparten el mismo vértice. En estos casos, las m posiciones serán tenidas en cuenta puesto que se trata de m posiciones diferentes entre sí, ya sea en cuanto al tipo de configuración, en cuanto al par de puntos con el que fueron obtenidas, etc, como vemos que ocurre en el ejemplo de la figura 5.16 para las posiciones con vértices en a_2 y a_5.

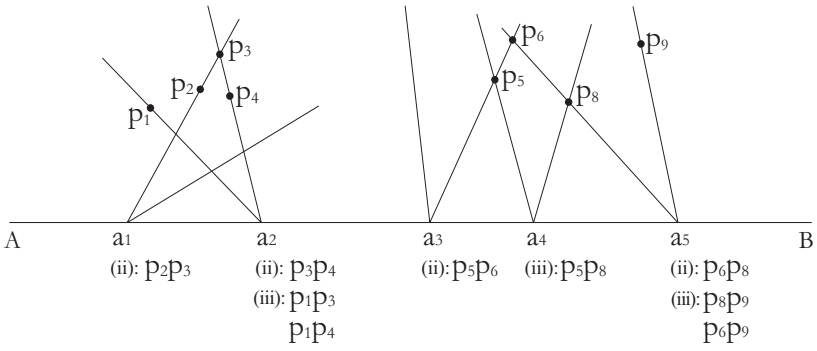


Figura 5.16: Algunas de las posiciones con configuración (ii) o (iii).

Veremos primero las consecuencias que puede traer no tener en cuenta los casos degenerados. De ahora en adelante hablaremos de "altura" del segmento en lugar de "posición" para dejar esta palabra reservada para la cuña.

1.) Más de dos puntos alineados:
Sabemos que tres puntos del conjunto de entrada alineados con una altura dan lugar al menos a tres posiciones para $\mathcal{C}$, puesto que en esta sección las posiciones están generadas por pares de puntos. En la figura 5.17 tenemos un ejemplo.

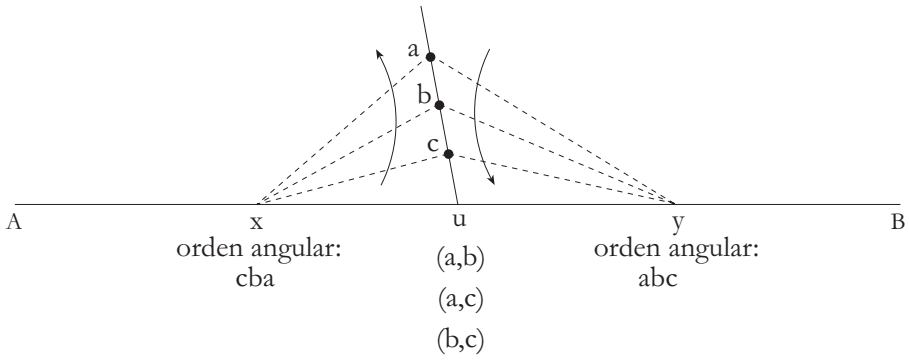
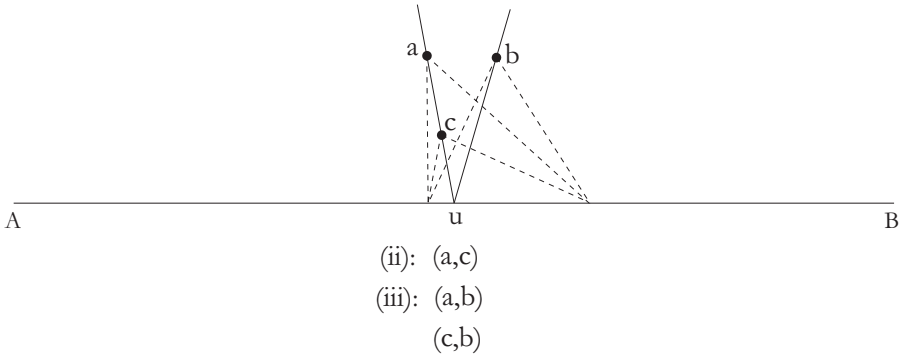


Figura 5.17: El caso degenerado.

Debajo de u están los pares de puntos asociados a las tres posiciones de $\mathcal{C}$ que tienen configuración (ii) y vértice en u. Estas posiciones, si las tratamos según la secuencia (a, c), (b, c) y (a, b), nos llevan a intercambiar primero el punto a con el c, luego el b con el c y finalmente el a con el b, con lo que el orden angular de los puntos termina como bca, que no es correcto. Esto quiere decir que las posiciones de $\mathcal{C}$, además de ser ordenadas a lo largo de $\overline{AB}$, también tendrán que ser ordenadas "a lo ancho", por decirlo de algún modo.

2.) Más de dos puntos en la frontera de una cuña:
Como vamos a trabajar con configuraciones de tipo (ii) y (iii) tenemos que ver ahora el orden relativo entre éstas en caso de que estén asociadas a una misma altura del segmento. La estrategia a seguir será analizar primero todas las configuraciones de un tipo y luego las de otro. La razón es que de no hacerlo así, la secuencia alterna de configuraciones de tipo (ii) y (iii) puede dar lugar a que

cada vez que llega la hora de contar el número de puntos cubiertos el orden de los puntos es tal que nunca están todos, como en el siguiente ejemplo.



El orden de los puntos al llegar a u es bca. Si la configuración (ii) se analiza entre las configuraciones de tipo (iii) de modo que el orden en que se trabaja con los pares de puntos es (c, b), (a, c) y (a, b) se tiene lo siguiente. Al estar (c, b) asociado a una configuración (iii), contamos el número de puntos que hay desde c hasta b y nos da 2. Luego viene el par (a, c), asociado a una configuración (ii), lo que implica intercambiar a y c en el orden, que se actualiza como bac. El último par de puntos es (a, b). Como están en configuración (iii), hay que contar, y el resultado es de nuevo 2. Al terminar, a, b y c quedaron ordenados de manera correcta sin embargo, el número máximo de puntos cubiertos dio 2, cuando en realidad es 3.

Finalmente, analizar primero todas las posiciones de configuración (ii) o primero todas las de configuración (iii) no es relevante. En un caso significa que actualizamos primero el orden de los puntos y luego contamos y en el otro caso, lo contrario. Como las configuraciones tienen en cuenta todas las combinaciones de puntos, a la hora de contar siempre habrá una combinación que dé la cantidad real de puntos cubiertos.

Veremos ahora el criterio para ordenar posiciones que tienen el mismo vértice. Sean entonces $((u_1, \beta_1), c_1, s_1, (p_1, q_1))$ y $((u_2, \beta_2), c_2, s_2, (p_2, q_2))$ dos posiciones cualesquiera. La norma a seguir va a ser que la configuración de tipo (ii) es

menor (se tratará antes) que la de tipo (iii) y que dentro de las posiciones con configuración (ii), aquellas donde los puntos estén en la semirrecta derecha se considerarán menores que aquellas donde los puntos estén en la semirrecta izquierda. Tenemos entonces que

$$((u_1,\beta_1),c_1,s_1,(p_1,q_1)) \quad \leq \quad ((u_2,\beta_2),c_2,s_2,(p_2,q_2))$$

$$\Longleftrightarrow$$

se cumple una de las siguientes condiciones:

$$u_1 < u_2,$$

$$u_1 = u_2 \quad y \quad \beta_1 < \beta_2,$$

$$u_1 = u_2, \quad \beta_1 = \beta_2 \quad y \quad c_1 < c_2,$$

$$u_1 = u_2, \quad \beta_1 = \beta_2, \quad c_1 = c_2 = (ii) \quad y \quad s_1 < s_2,$$

$$u_1 = u_2, \quad \beta_1 = \beta_2, \quad c_1 = c_2 = (ii), \quad s_1 = s_2 \quad y \quad p_1 < p_2,$$

$$u_1 = u_2, \quad \beta_1 = \beta_2, \quad c_1 = c_2 = (ii), \quad s_1 = s_2, \quad p_1 = p_2 \quad y \quad q_1 < q_2.$$

Como es fácil comprobar, ni el algoritmo ni su complejidad se ven alterados cuando tenemos en cuenta las degeneraciones ya que los casos degenerados sólo afectan el paso [3] del algoritmo: hallar el orden en que las posiciones van a ser tratadas. El tratamiento dado a cada posición en el paso [5] sigue siendo el mismo.

5.4.2 Algoritmo de las Franjas

En esta sección veremos otro algoritmo para hallar la solución dentro del conjunto de posiciones en configuración cuadrática. De manera general, se trata de un algoritmo bastante sencillo cuyos únicos dos pasos son construir ciertos conjuntos e intersectarlos. La solución estará en donde se intersecte el mayor número de conjuntos. Como veremos, los dos pasos pueden implementarse con relativa facilidad dando lugar a un algoritmo de complejidad $O(n^2\,\alpha(n))$, o pueden implementarse utilizando técnicas un poco más sofisticadas, en cuyo caso la complejidad del algoritmo baja hasta $O(n^2)$.

La idea general es hallar, por cada punto de entrada p, el conjunto de todas las posiciones donde $\mathcal{C}$ cubre a p. Haciendo la intersección de estos conjuntos podemos determinar las posiciones donde se cubre la mayor cantidad de puntos, que no son otras que las posiciones que pertenecen a las intersecciones donde interviene el mayor número de conjuntos. En la figura 5.18 tenemos un ejemplo. Hay dos conjuntos, cada uno inducido por un punto. La región sombreada es la que contiene las posiciones que componen la solución.

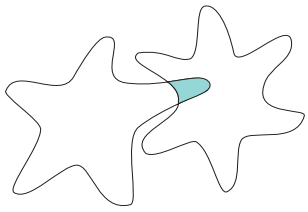


Figura 5.18: Intersección de dos conjuntos.

Para hallar la solución seguimos considerando que el segmento $\overline{AB}$ está sobre el eje `OX`, que A coincide con el origen de coordenadas y que los puntos de entrada están estrictamente por encima de $\overline{AB}$. La situación general (que haya puntos sobre el segmento) se analiza posteriormente, ya veremos por qué.

El conjunto inducido por un punto lo vamos a obtener determinando, para cada posición del vértice de la cuña, los valores extremos del intervalo dentro del cual puede rotar $\mathcal{C}$ para que el punto quede cubierto.

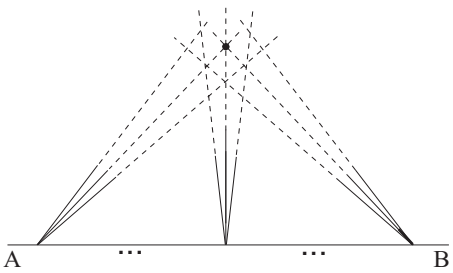
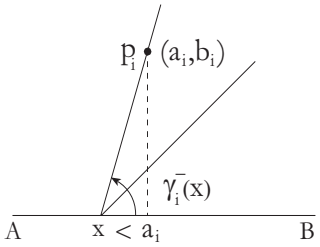


Figura 5.19: Cubriendo al punto.

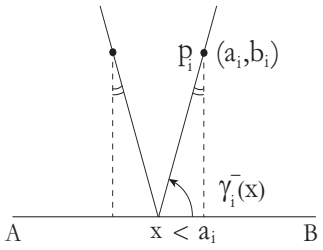
El vértice de $\mathcal{C}$ puede ser cualquier punto de $\overline{AB}$. Los valores del intervalo de rotación se dan con respecto a la semirrecta izquierda de $\mathcal{C}$. Construyamos para cada punto p_i una función que, dado el valor x del desplazamiento del vértice, devuelve el valor que tiene que tomar β para que en la posición (x, β) la semirrecta izquierda de $\mathcal{C}$ pase por p_i. Llamamos $\gamma_i^-(x)$ a esta función y la definimos del siguiente modo:

$$\gamma_i^-(x) = \begin{cases} \arctan\left(\frac{b_i}{a_i - x}\right) & \text{si } x < a_i \\ \frac{\pi}{2} & \text{si } x = a_i \\ \pi - \arctan\left(\frac{b_i}{x - a_i}\right) & \text{si } x > a_i \end{cases}$$



Sin embargo, en vez de utilizar esta parametrización utilizaremos la que tenemos a continuación, por ser más uniforme.

$$\gamma_i^-(x) = \frac{\pi}{2} - \arctan\left(\frac{a_i - x}{b_i}\right)$$



La nueva definición facilita el operar con las curvas sin embargo, los puntos no pueden estar en $\overline{AB}$ ya que en este caso b_i, en el denominador, sería igual a cero. Como ya dijimos, los puntos que pertenezcan a $\overline{AB}$ serán analizados al final.

La función $\gamma_i^-(x)$ nos da entonces el extremo izquierdo del intervalo de rotación. El extremo derecho lo obtenemos haciendo rotar $\mathcal{C}$ en sentido antihorario hasta que la semirrecta derecha choque con p_i. Llamamos $\gamma_i^+(x)$ a la función que nos da el extremo derecho del intervalo de rotación. Como $\mathcal{C}$ es de ángulo α podemos escribir $\gamma_i^+(x)$ del siguiente modo:

$$\gamma_i^+(x) = \gamma_i^-(x) + \alpha$$

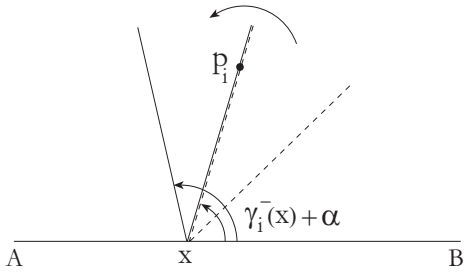


Figura 5.20: Geométricamente.

Si representamos $\gamma_i^-(x)$ y $\gamma_i^+(x)$ en el plano vemos que cada una es una traslación de la otra α unidades en el eje OY. La región cerrada comprendida entre estas dos curvas contiene al conjunto de posiciones del que hablábamos al inicio: el conjunto de posiciones donde $\mathcal{C}$ cubre a p_i. A estas regiones las llamaremos *franjas*. La definición de una franja es entonces la siguiente:

$$\mathcal{F}_i = \{(x, y) \ / \ \gamma_i^-(x) \leq y \leq \gamma_i^+(x)\}$$

En la figura 5.21 tenemos un ejemplo. En esta figura, cuando el vértice de la cuña está en a, $\gamma_i^-(a)$ y $\gamma_i^+(a)$ son los valores extremos del intervalo dentro del cual puede moverse la semirrecta izquierda de $\mathcal{C}$ de modo que p_i quede dentro de $\mathcal{C}$.

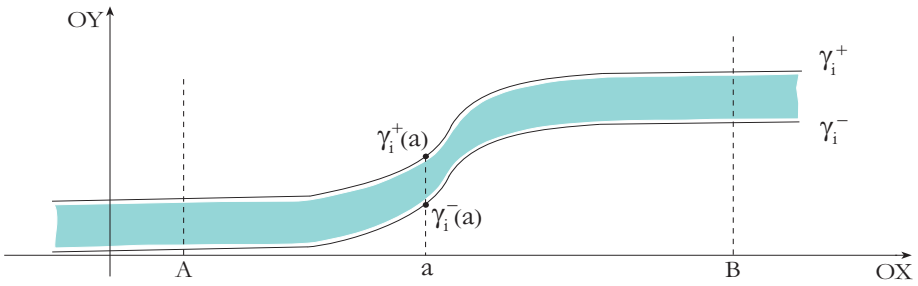


Figura 5.21: Franja generada por un punto.

Para hallar la región del plano donde se intersecta el mayor número de franjas construimos el arreglo de las curvas que las definen. Veamos primero el comportamiento de estas curvas.

Propiedades de las Curvas

Sea Γ el conjunto $\{\gamma_1^-(x), \gamma_1^+(x), \gamma_2^-(x), \gamma_2^+(x), \ldots, \gamma_n^-(x), \gamma_n^+(x)\}$. Como veremos a continuación, en el problema que nos ocupa las curvas de Γ se intersectan en a lo sumo dos puntos y hay pares de curvas que pueden ser tangentes entre sí.

Propiedad 5.4.5. *Las curvas de* Γ *se intersectan en a lo sumo dos puntos.*

Demostración. Sean p_i y p_j dos puntos cualesquiera del conjunto de entrada. Construyamos la circunferencia que contiene al arco capaz de ángulo α que pasa por p_i y p_j. Como ya sabemos, las intersecciones entre esta circunferencia y el segmento $\overline{AB}$ nos dan las posiciones (u, β) donde $\mathcal{C}$ tiene a p_i en una semirrecta y a p_j en la otra. Supongamos, sin pérdida de generalidad, que en estas posiciones p_i está en la semirrecta izquierda y p_j pertenece a la semirrecta derecha. Hecha esta suposición y teniendo en cuenta las definiciones de γ^- y γ^+ sabemos que las posiciones (u, β) se traducen, en el contexto de las franjas, en puntos que pertenecen tanto a γ_i^- como a γ_j^+. Consecuencia de esto es que el número de posiciones (u, β) coincide con el número de veces que γ_i^- y γ_j^+ se intersectan entre sí. La figura 5.22 es un ejemplo.

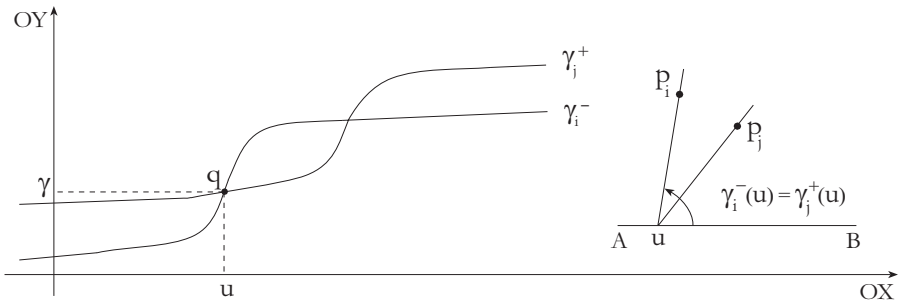


Figura 5.22: La intersección doble entre dos curvas.

Sabiendo que el número de posiciones (u, β) es a lo sumo dos (las posiciones (u, β) son el resultado de la intersección de una circunferencia -la circunferencia soporte del arco capaz- y un segmento -el segmento $\overline{AB}$-) llegamos a lo que queríamos demostrar.

□

De la demostración anterior se deduce que en el arreglo $\mathcal{A}(\Gamma)$ podemos encontrar curvas que no se intersecten entre sí, otras que lo hagan en un punto y otras que lo hagan en dos. Veremos ahora que cuando las curvas se intersectan en un punto lo hacen de manera tangente.

Propiedad 5.4.6. *Cuando una curva superior de* Γ *tiene un solo punto de intersección con una curva inferior de* Γ*, este punto es un punto de tangencia.*

Demostración. Sea (u, β) el único punto de intersección entre las curvas γ_i^- y γ_j^+. Probaremos que el orden entre estas curvas es el mismo tanto a la izquierda como a la derecha de u. Como ya sabemos de la demostración anterior, el punto u de $\overline{AB}$ es un punto de tangencia entre $\overline{AB}$ y la circunferencia $\mathcal{D}$ que contiene al arco capaz de ángulo α que pasa por p_i y p_j.

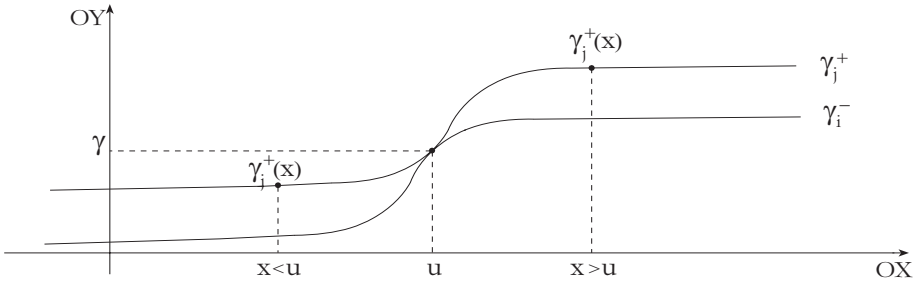


Figura 5.23: Intersección no transversal entre dos curvas.

Sea x un punto cualquiera a la izquierda de u. Si trazamos las semirrectas que parten de x y pasan por p_i y p_j vemos que éstas forman un ángulo menor que α, por encontrarse x fuera de $\mathcal{D}$. Figura 5.24.

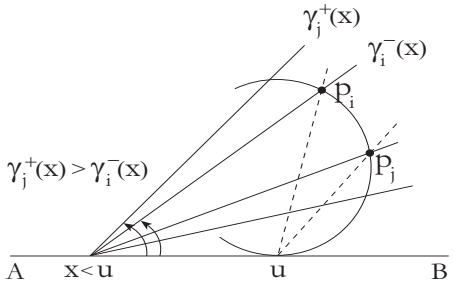


Figura 5.24: El orden angular entre las semirrectas izquierdas no se altera al pasar por u.

Para que el ángulo entre las semirrectas alcance el valor α mientras el punto de partida de las mismas está fijo en x y la semirrecta derecha pasa por p_j tenemos que hacer rotar la semirrecta izquierda en sentido antihorario. Esto quiere decir que cuando $\mathcal{C}$ tenga su vértice en x, la curva γ_j^+ estará por encima de γ_i^-. Para las posiciones x a la derecha de u ocurre lo mismo. En la figura 5.25 tenemos un dibujo para este caso $x > u$.

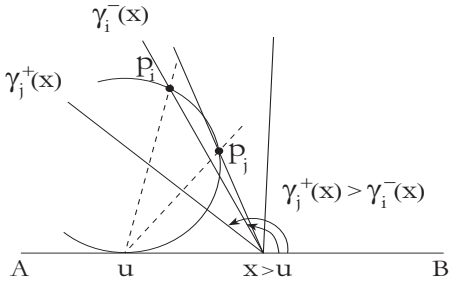


Figura 5.25: El orden angular entre las semirrectas izquierdas no se altera al pasar por u.

□

Construcción del Arreglo de Curvas

El conjunto Γ es un conjunto de curvas de arcotangente que se intersectan dos a dos en a lo sumo dos puntos y en el que pueden existir curvas tangentes entre sí. El Teorema 3.11.1 nos dice que en este caso el arreglo $\mathcal{A}(\Gamma)$ puede ser construido de manera incremental en tiempo $O(n^2\,\alpha(n))$.

Etiquetado de las Caras del Arreglo

Una vez que tenemos el arreglo, el último paso es etiquetar las caras con el número de franjas a las que pertenecen. Para esto utilizamos una estrategia *voraz*. La idea es partir de una cara cualquiera $\mathcal{F}$, etiquetarla, y visitar de una en una todas las caras que le son adyacentes y no han sido aún etiquetadas, para hacer lo mismo. Este método es recursivo y termina cuando todas las caras adyacentes a la cara de partida han sido visitadas. Como bien sabemos, este etiquetado puede hacerse en tiempo cuadrático.

En este caso se puede empezar por la cara exterior, que se sabe tiene etiqueta igual a cero. La cara exterior se puede reconocer fácilmente ya que es la cara adyacente a la envolvente inferior y a la envolvente superior del arreglo.

La etiqueta de las caras adyacentes a una cara $\mathcal{F}$ será igual a la etiqueta de $\mathcal{F}$ más o menos una unidad. Se sumará una unidad si al pasar de $\mathcal{F}$ a la cara se entra en una franja y se restará una unidad si al pasar de $\mathcal{F}$ a la cara se sale de una franja.

Se sale de una franja si la arista por la que se pasa de una cara a la otra pertenece a una curva superior y al recorrer esta curva en sentido creciente la cara de llegada está a la izquierda de la curva. También se sale cuando se trata de una curva inferior y al recorrerla en sentido creciente la cara de llegada queda a la derecha de la curva. De manera análoga se puede determinar cuándo se entra en una franja. La información sobre si una cara está a la izquierda o la derecha de una arista puede obtenerse del arreglo.

Durante todo el proceso se llevará la cuenta de las caras con etiqueta máxima y del valor de esta etiqueta máxima. Todos los puntos de las caras con etiqueta máxima formarán parte de la solución. El valor de la etiqueta de esta cara será el número de puntos cubiertos.

Pasos del Algoritmo

Algoritmo de las Franjas

[1] Por cada punto p_i, $1 \leq i \leq n$, construir las curvas γ_i^- y γ_i^+ que definen la franja correspondiente.

[2] Construir el arreglo $\mathcal{A}$ ($\{\gamma_1^-, \gamma_1^+, \gamma_2^-, \gamma_2^+, \ldots, \gamma_n^-, \gamma_n^+\}$).

[3] Etiquetar las caras del arreglo con el número de franjas a las que pertenecen.

[4] Devolver las caras de etiqueta máxima.

Con todo lo visto hasta aquí podemos decir que el algoritmo de las franjas tiene complejidad $O(n^2\,\alpha(n))$. Por las características de la función $\alpha(n)$, un algoritmo de complejidad $O(n^2\,\alpha(n))$ puede considerarse como cuadrático a efectos prácticos; sin embargo, desde un punto de visto teórico, el algoritmo $O(n^2\,\alpha(n))$ tiene complejidad mayor que cualquier algoritmo $O(n^2)$. Veremos ahora que el arreglo de curvas de arcotangente puede ser construido en tiempo cuadrático si utilizamos el algoritmo de Balaban [2]. Esto reducirá la complejidad del algoritmo de las franjas a $O(n^2)$.

Utilizando el Algoritmo de Balaban.

Otra variante para la construcción del arreglo de franjas es utilizar el algoritmo de *Balaban*. La implementación de este algoritmo es un poco trabajosa, sin embargo, la complejidad del tiempo de construcción del arreglo es cuadrática, como veremos a continuación.

Balaban en su artículo presenta dos algoritmos para hallar las intersecciones de un conjunto de segmentos de recta. Los algoritmos utilizan espacio lineal y tienen complejidad $O(n\,\log^2(n) + k)$ y $O(n\,\log(n) + k)$ en cuanto al tiempo, siendo k el número de intersecciones (asumiendo que todos los puntos de intersección son diferentes). Los segmentos por su parte pueden ser arcos de curva o cualquier

objeto $2D$ conexo con a lo sumo una intersección con cualquier recta vertical y con los cuales podamos hacer las siguientes operaciones en tiempo $O(1)$:

- Dados un arco y una recta vertical, hallar el punto de intersección.
- Dados un par de arcos y una banda, determinar si los arcos se intersectan dentro de la banda.

Nuestras curvas cumplen con los requisitos que hay que cumplir para poder ejecutar este algoritmo. Primero, son funciones de *arcotangente* con imagen en el intervalo $(0..\pi)$. Segundo, la intersección con cualquier recta vertical r de ecuación $x = c$ es única y puede ser hallada en tiempo constante:

$$\gamma_i^-(x) \cap r = \left(c, \frac{\pi}{2} - \arctan(\frac{a_i - c}{b_i})\right)$$

Determinar si dos curvas γ_i y γ_j se intersectan dentro de una banda requiere también tiempo constante. En caso de que las dos curvas sean inferiores o las dos sean superiores, hallar la abscisa del punto de intersección sería despejar la variable x en una ecuación del siguiente tipo:

$$\frac{a_i - x}{b_i} = \frac{a_j - x}{b_j}$$

En caso de tratarse de una curva superior y otra inferior la intersección se hallaría en tiempo también constante. A continuación tenemos los pasos para despejar x en la igualdad $\gamma_i^- = \gamma_j^+$. Para pasar de la tercera ecuación a la cuarta hemos utilizado la igualdad $\tan(\gamma - \phi) = \frac{\tan\gamma - \tan\phi}{1 + \tan\gamma\tan\phi}$. Obsérvese que para obtener el valor de x sólo es preciso resolver una ecuación de segundo grado.

$$\arctan\left(\frac{a_i - x}{b_i}\right) = \arctan\left(\frac{a_j - x}{b_j}\right) + \alpha$$

$$\arctan\left(\frac{a_i - x}{b_i}\right) - \arctan\left(\frac{a_j - x}{b_j}\right) = \alpha$$

$$\tan\left(\arctan\left(\frac{a_i - x}{b_i}\right) - \arctan\left(\frac{a_j - x}{b_j}\right)\right) = \tan\alpha$$

$$\frac{a_i - x}{b_i} - \frac{a_j - x}{b_j} = \tan\alpha \ \left(1 + \frac{(a_i - x)(a_j - x)}{b_i b_j}\right)$$

$$b_j(a_i - x) - b_i(a_j - x) = \tan\alpha \ (b_i b_j + (a_i - x)(a_j - x))$$

Como en nuestro caso el número de intersecciones diferentes entre las curvas puede ser de orden cuadrático, llegamos a que el arreglo de curvas de arcotangente puede construirse en tiempo cuadrático utilizando el algoritmo de Balaban.

Análisis de la Complejidad

Con todo lo visto hasta ahora podemos enunciar el siguiente lema.

Lema 5.4.2. *El algoritmo de las Franjas tiene complejidad* $O(n^2)$.

Casos Degenerados

Quedó pendiente de análisis el caso de puntos que pertenecieran al segmento ya que para estos puntos no se definieron las franjas.

Sea $\mathcal{H}$ una de las caras del arreglo de etiqueta máxima k. Esto quiere decir que cada uno de los puntos que pertenece a esta cara es una posición sobre el segmento donde $\mathcal{C}$ cubre k puntos del conjunto de entrada. Si en alguna de estas posiciones se cubre también un punto que pertenece al segmento y que, por lo tanto, no fue tenido en cuenta, el tenerlo en cuenta ahora sólo hace aumentar en uno la cantidad de puntos cubiertos por la cuña.

De las posiciones donde el punto $p \in \overline{AB}$ es cubierto por $\mathcal{C}$ sólo nos interesan

aquellas en las que $\mathcal{C}$ tiene su vértice en p ya que las restantes pueden reducirse a la configuración (i). Si (a, b) son las coordenadas de p entonces la posición donde $\mathcal{C}$ tiene su vértice en p y el ángulo es arbitrario viene representada en nuestro sistema de coordenadas por la recta vertical $x = a$. Intersectando $\mathcal{H}$ y la recta $x = a$ obtenemos todas las posiciones donde $\mathcal{C}$ cubre $k+1$ puntos. En caso de que haya varios puntos sobre el segmento, el tenerlos en cuenta puede hacer aumentar en sólo una unidad el número de puntos cubiertos ya que el vértice de la cuña sólo puede estar en uno de estos puntos.

Finalmente, la estrategia a seguir cuando se tienen en cuenta los casos degenerados es la siguiente. Se siguen todos los pasos vistos en el algoritmo y al final se hallan las intersecciones entre las caras $\mathcal{H}$ del conjunto solución y las rectas verticales, correspondiente cada una a un punto sobre el segmento. Si no se da ninguna intersección entre las caras y las rectas, el conjunto solución sigue siendo el mismo. En caso de darse al menos una intersección el nuevo conjunto solución está formado por la unión de todas las nuevas intersecciones (segmentos de recta) y la cantidad máxima de puntos cubiertos pasa de k a $k + 1$.

Todos los segmentos de rectas que componen la nueva solución pueden hallarse en tiempo $O(n^2)$ haciendo un recorrido del arreglo por las curvas de nivel. Las curvas de nivel se recorrerán de izquierda a derecha, comenzando por la de nivel cero que es en nuestro caso la que coincide con la envolvente superior, según tenemos en el capítulo de Técnicas de Implementación. El recorrido de cada curva se hará en paralelo con la visita de las rectas, que habrán sido ordenadas también de izquierda a derecha. Para cada arista de una curva se preguntará si alguna de sus dos caras adyacentes pertenece al conjunto solución (bastará con preguntar si la cara es de etiqueta máxima). En caso de serlo, se hallarán las intersecciones entre la arista y las rectas.

A excepción de las caras por encima de la envolvente superior del arreglo y por debajo de la envolvente inferior, toda cara que pertenezca a la solución va a ser adyacente a dos curvas de nivel. Saber esto facilita la obtención de los segmentos de recta que estamos buscando: una vez que hemos encontrado k extremos de

estos segmentos en una curva (o sea, intersecciones entre una cara solución y una recta vertical) sabemos que otros k extremos aparecerán en la siguiente curva. Enlazando estos extremos por pares vamos obteniendo los segmentos de recta solución.

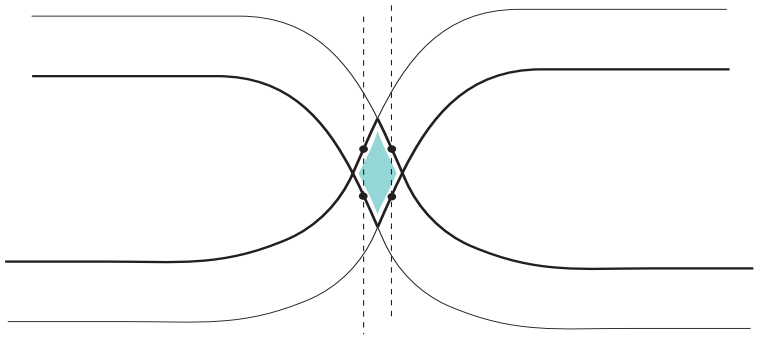


Figura 5.26: Las curvas de nivel y la nueva solución.

5.4.3 Algoritmo de Paso al Dual

En esta sección vamos a ver que utilizando la técnica del Dual obtenemos también un algoritmo de complejidad cuadrática sin necesidad de construir arreglos de manera trabajosa. Veremos que en este caso todos los arreglos que aparezcan podrán ser construidos en tiempo cuadrático utilizando el método incremental. El conjunto solución, sin embargo, vuelve a ser ahora un conjunto discreto.

Consideraremos, sin pérdida de generalidad, que el segmento $\overline{AB}$ está en el semieje `OY` positivo, coincidiendo A con el origen de coordenadas, y que los puntos se encuentran todos a la derecha de $\overline{AB}$.

La idea general del algoritmo es pasar al Dual convirtiendo el conjunto de puntos en un conjunto de rectas y las semirrectas de $\mathcal{C}$ en un par de puntos. Una vez en el Dual se construye el arreglo que las rectas duales generan y es en el arreglo donde se busca la solución. La transformación que utilizamos para pasar al Dual es la transformación $\mathcal{T}$ vista en el capítulo de Técnicas de Implementación. Como recordaremos, las propiedades de $\mathcal{T}$ no van a cumplirse para semirrectas verticales, sin embargo, que el segmento sea ahora vertical trae como consecuencia que

aquellas posiciones de $\mathcal{C}$ en las que una de las semirrectas es vertical no tienen que ser tratadas, ya que pueden ser llevadas a posiciones de configuración (i).

Nos referiremos entonces a las semirrectas de $\mathcal{C}$ como superior e inferior, en lugar de izquierda y derecha respectivamente. Veamos ahora cómo expresar la posición de la cuña en el problema Dual y luego cómo obtener el número de puntos que se cubren en cada posición.

La Posición de $\mathcal{C}$ en el Problema Dual

Denotemos por $\mathcal{A}^*$ al arreglo de rectas en el Dual. Teniendo en cuenta la observación 2.9.1 y el hecho de que en la configuración (ii) una de las semirrectas de $\mathcal{C}$ pasa por al menos dos puntos vemos que todas las posiciones del problema Primal donde $\mathcal{C}$ tiene configuración (ii) están presentes en el Dual como uno de los vértices de $\mathcal{A}^*$

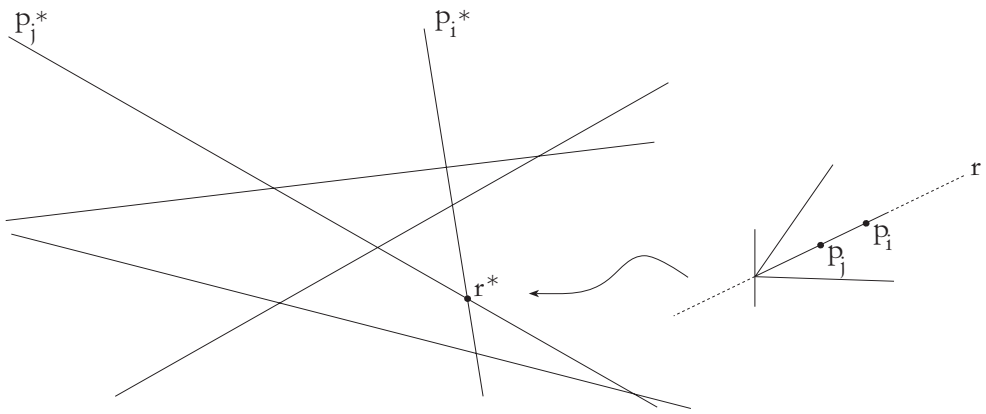


Figura 5.27: Las cuñas de configuración (ii) y el arreglo.

Como gracias a la propiedad 5.4.1 podemos prescindir de las posiciones primales de configuración (iii) vemos que visitando los vértices del arreglo (donde están las posiciones de configuración (ii)) estamos visitando todas las posiciones entre las que se encontrará la solución. En otras palabras,

Observación 5.4.2. Las posiciones candidatas a solución están localizadas en el

arreglo $\mathcal{A}^*$ explícitamente.

Ahora hay que tener en cuenta que en el problema Primal cada una de las posiciones de configuración (ii) puede dar lugar a dos situaciones distintas: una en la que la semirrecta que pasa por al menos dos puntos es la inferior y otra en que la que pasa por al menos dos puntos es la superior. Esto quiere decir que para buscar la solución en el problema Dual hay que considerar dos fases.

Observación 5.4.3. Los vértices del arreglo $\mathcal{A}^*$ van a ser considerados en una primera fase como los transformados de semirrectas inferiores y en una segunda, como los transformados de semirrectas superiores.

Ambos casos son simétricos. De ahora en adelante y para facilitar la explicación supondremos que los vértices del arreglo son los transformados de rectas soporte de semirrectas inferiores. Hasta aquí lo referente a la transformación que hace $\mathcal{T}$ de las posiciones primales candidatas a solución. Nos ocupamos ahora de cómo obtener en el Dual el número de puntos cubiertos por $\mathcal{C}$ en una determinada posición.

$|\mathcal{C}|$ en el Problema Dual

La cantidad de puntos cubiertos por $\mathcal{C}$ en el Primal puede obtenerse a partir de la cantidad de puntos que se hayan por encima de cada una de las semirrectas, como vemos en la figura 5.28.

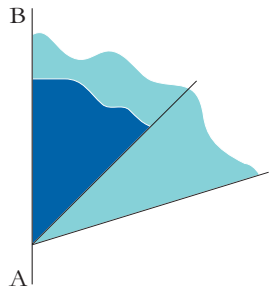


Figura 5.28: Ilustración.

En particular, al número de puntos que están por encima de la semirrecta inferior o que pertenecen a ella hay que restarle el número de puntos que están estrictamente por encima de la semirrecta superior. Por la propiedad 2.9.2 de $\mathcal{T}$, la cantidad de puntos que están en el Primal estrictamente por encima de una semirrecta r es igual en el Dual a la cantidad de rectas que están estrictamente por debajo del punto r^* (el transformado de r).

Sean r y s las semirrectas inferior y superior de $\mathcal{C}$ respectivamente. El cálculo de $|\mathcal{C}|$ en el Dual se hace entonces sumándole al número de rectas que pasan por r^* el valor absoluto de la diferencia entre el número de rectas que pasan por debajo de r^* y el número de rectas que pasan por debajo de s^*. El primer sumando se refiere al número de puntos que pertenecen a r en el Primal. En la figura 5.29 vemos un ejemplo.

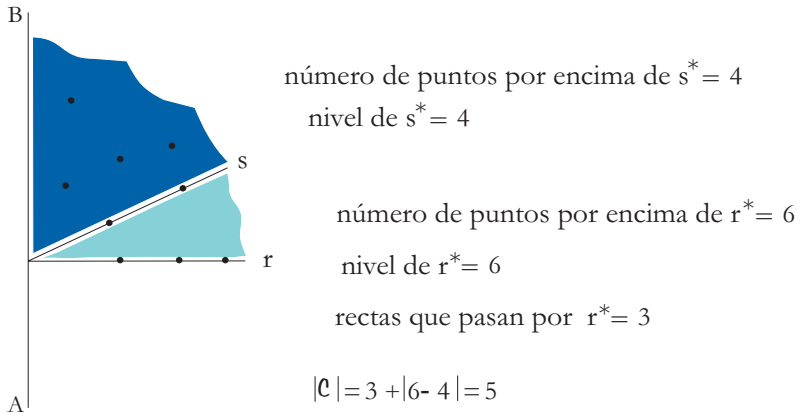


Figura 5.29: Calculando el número de puntos cubiertos.

Ahora tenemos en cuenta que r^* es un vértice del arreglo pero s^* no tiene por qué serlo ya que corresponde en el Primal a la semirrecta superior de $\mathcal{C}$ que en las posiciones de configuración (ii) que estamos tratando (los puntos generadores están en la semirrecta inferior) no está obligada a pasar por ningún punto. Por lo tanto,

- el número de rectas del arreglo que pasan por debajo de r^* puede obtenerse del arreglo a partir de la curva de nivel a la que r^* pertenece;

- para obtener el número de rectas del arreglo que pasan por debajo de s^* tenemos que localizar s^* dentro del arreglo.

El etiquetado de los elementos del arreglo (vértices, aristas y caras) con la curva de nivel a que pertenece puede hacerse en tiempo cuadrático, como vimos en el capítulo de Técnicas de Implementación. Una vez etiquetado el arreglo, el número de rectas por debajo de r^* se obtiene en tiempo constante. Nos ocupamos entonces de la localización de s^* dentro del arreglo.

s^* y el Arreglo

Localizar s^* en $\mathcal{A}^*$ quiere decir hallar el vértice, arista o cara a la que s^* pertenece. El número de rectas que pasen por debajo de s^* será igual al nivel de dicho vértice, arista o cara.

Como $\overline{AB}$ pertenece a `OY` las semirrectas de $\mathcal{C}$ se intersectan con este eje a la misma altura, digamos n. Gracias a la forma en que se elegió $\mathcal{T}$, las semirrectas r y s se convierten en el Dual en los puntos de coordenadas $(m_r, -n)$ y $(m_s, -n)$ respectivamente o sea, dos puntos que se encuentran en el arreglo a la misma altura $-n$, siendo $(m_r, -n)$ un vértice del arreglo. En la figura 5.30 tenemos un ejemplo. En esta figura aparecen dibujados los asociados de algunos de los vértices del arreglo.

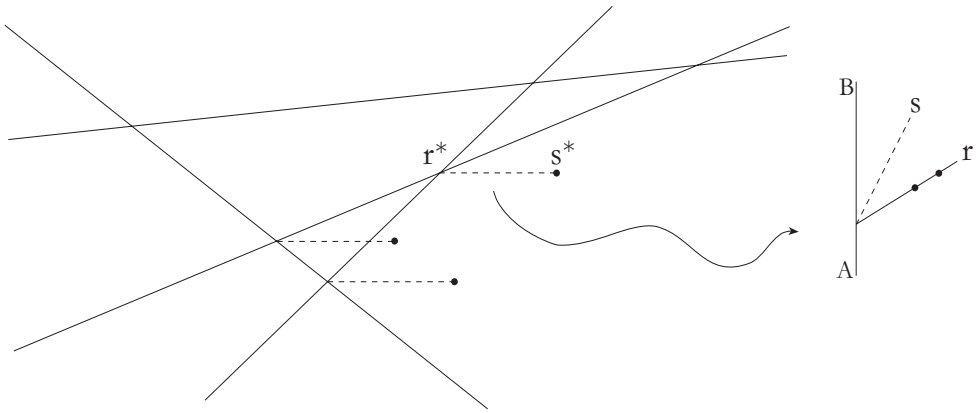


Figura 5.30: Vértices y sus puntos asociados.

Definición 5.4.1. *Llamamos* punto asociado a un vértice *al transformado de la semirrecta superior de la cuña cuya semirrecta inferior corresponde a dicho vértice.*

Localicemos, para cada vértice r^* del arreglo, su punto asociado s^*.

Vértices y Puntos Asociados

Sea p^* una recta cualquiera del arreglo. Hagamos corresponder a cada punto de p^* su asociado. El conjunto de todos los asociados da lugar a una curva, como vemos en la figura 5.31. Insertando esta curva en $\mathcal{A}^*$ localizamos de manera inmediata los asociados a los vértices de p^* ya que todo vértice y su asociado se encuentran a la misma altura.

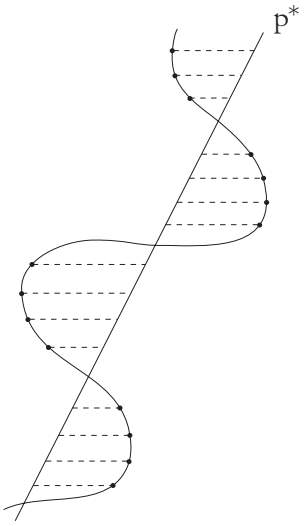


Figura 5.31: Puntos de p^* y sus asociados.

De manera general, se definirá una curva por cada recta. Las curvas se insertarán en el arreglo de una en una y serán inmediatamente borradas una vez localizados los puntos asociados correspondientes a los vértices.

Definición 5.4.2. *Llamamos* curva asociada a una recta p^* *a la curva que pasa por los asociados a los puntos que pertenecen a* p^*. *Utilizaremos la notación* $c(p^*)$

para referirnos a la curva asociada a p^*.

Ecuación de la Curva Asociada

Para hallar la ecuación de $c(p^*)$ hallaremos la relación existente entre las abscisas y las ordenadas de los puntos de $c(p^*)$. Obtendremos así la ecuación implícita de la curva.

Sea entonces (m_{sup}, n_{sup}) un punto cualquiera de $c(p^*)$. La notación utilizada para el punto nos recuerda que en el Primal m_{sup} y n_{sup} son los parámetros de la recta soporte de la semirrecta superior de $\mathcal{C}$ para la posición en cuestión. Por ser (m_{sup}, n_{sup}) un punto de $c(p^*)$, es el asociado al punto de p^* que se encuentra a su misma altura. Sean (m_{inf}, n_{inf}) las coordenadas de este punto de la recta. Por pertenecer a la recta, (m_{inf}, n_{inf}) satisface la ecuación de la misma. Sea $y = ax - b$ la ecuación de p^*. Tenemos entonces que $n_{inf} = am_{inf} - b$. Por estar cada punto y su asociado a la misma altura, se cumple que $n_{sup} = n_{inf}$, luego,

$$n_{sup} = a\, m_{inf} - b \qquad (1)$$

Si ponemos m_{sup} en función de m_{inf}, tendremos ya la relación entre las componentes de los puntos de la curva y el parámetro m_{inf} de la recta p^*.

Sean δ_{inf} y δ_{sup} respectivamente los ángulos que forman las semirrectas inferior y superior de $\mathcal{C}$ con la prolongación positiva del eje `OX` en la posición que estamos tratando. Debido a que nos dedicamos por el momento a semirrectas no verticales, δ_{inf} y δ_{sup} van a ser diferentes de $-\frac{\pi}{2}$ y $\frac{\pi}{2}$ respectivamente. Esto quiere decir que las rectas soporte de las semirrectas van a tener pendiente definida. Se cumple entonces que $m_{inf} = \tan(\delta_{inf})$ y $m_{sup} = \tan(\delta_{sup})$. Dado que $\mathcal{C}$ es de ángulo fijo,

$$\delta_{sup} - \delta_{inf} = \alpha$$

$$\tan(\delta_{sup} - \delta_{inf}) = \tan(\alpha)$$

$$\frac{\tan(\delta_{sup}) - \tan(\delta_{inf})}{1 + \tan(\delta_{sup})\tan(\delta_{inf})} = \tan(\alpha)$$

$$\frac{m_{sup} - m_{inf}}{1 + m_{sup}m_{inf}} = \tan(\alpha)$$

$$m_{sup} - m_{inf} - \tan(\alpha)(1 + m_{sup}m_{inf}) = 0$$

$$m_{sup}(1 - m_{inf}\tan(\alpha)) - m_{inf} - \tan(\alpha) = 0$$

$$m_{sup} = \frac{m_{inf} + \tan(\alpha)}{1 - m_{inf}\tan(\alpha)} \qquad (2)$$

Las ecuaciones (1) y (2) nos dan la parametrización de la curva $c(p^*)$.

$$c(p^*) = \left\{ \left(\frac{t + \tan(\alpha)}{1 - t\ \tan(\alpha)},\ a\,t - b \right)\ / \ t \in \mathbb{R} - \{tan^{-1}(\alpha)\} \right\}$$

De esta expresión pasamos a la ecuación implícita despejando el parámetro t en la primera componente, x, y sustituyéndolo en la segunda, y. La ecuación implícita de la curva es entonces la siguiente:

$$y = a\frac{x - \tan(\alpha)}{1 + x\ \tan(\alpha)} - b. \qquad (3)$$

Esta ecuación tiene sentido siempre y cuando α sea diferente de $\frac{\pi}{2}$. Para $\alpha = \frac{\pi}{2}$, $\tan(\alpha)$ no está definida, sin embargo, las rectas soporte de $\mathcal{C}$ son en este caso perpendiculares entre sí y la relación entre m_{inf} y m_{sup}, pendientes de estas rectas, es inmediata, siendo $m_{sup} = -\frac{1}{m_{inf}}$. La ecuación de la curva es entonces la siguiente:

$$y = -\frac{a}{x} - b \qquad (4)$$

Esta misma ecuación se obtiene si se hace el límite cuando x tiende a $\frac{\pi}{2}$ tanto por la derecha como por la izquierda. Pasemos ahora al análisis de la curva ya que nuestro objetivo es insertarla en $\mathcal{A}^*$. Un argumento sencillo nos permite afirmar que se trata de una hipérbola. En efecto, la ecuación (3) es una ecuación cuadrática en las variables x e y de lo que se deduce que estamos ante una cónica. Si ahora observamos que dicha cónica tiene una asíntota vertical en $x = -tan^{-1}(\alpha)$ podemos concluir que la cónica es una hipérbola.

Análisis de la Curva

Analizaremos solamente el caso en que la ecuación de la curva es (3), ya que para (4) la curva se comporta de manera similar. Nuestra curva es entonces

$$y = a\frac{x - \tan(\alpha)}{1 + x\tan(\alpha)} - b.$$

Esta ecuación corresponde a una función que tiene como dominio el conjunto $\mathbb{R} - \{\frac{-1}{\tan(\alpha)}\}$, por ser $\frac{-1}{\tan(\alpha)}$ el valor donde se anula el denominador. En $x = \frac{-1}{\tan(\alpha)}$ hay, por lo tanto, una asíntota vertical. Existe también una asíntota horizontal en $y = \frac{a}{\tan(\alpha)} - b$.

No hay valores extremos ni puntos de inflexión. A la izquierda de la asíntota vertical la función es cóncava y a la derecha convexa, puesto que a es siempre mayor o igual que cero. La curva es también monótona. Por último, la intersección con el eje `OX` se da en un único punto, lo mismo que con el eje `OY`. Con lo visto hasta aquí podemos ya esbozar la gráfica de la función, que aparece representada en la figura 5.32.

Como es fácil comprobar, la intersección de esta curva con una recta cualquiera del arreglo puede darse en a lo sumo dos puntos. Veamos ahora la interpretación

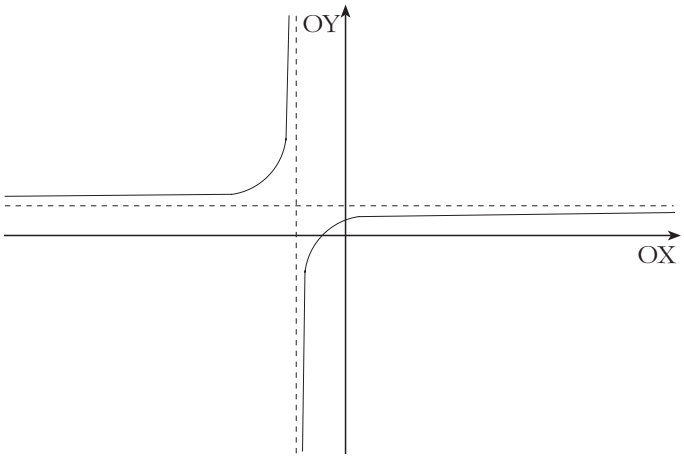


Figura 5.32: Gráfica de la curva.

geométrica de las asíntotas.

(a) Asíntotas Horizontales:
Sea $y = \frac{a}{\tan(\alpha)} - b$ la ecuación de la asíntota horizontal de la curva $c(p^*)$. Llamemos y_0 al valor $\frac{a}{\tan(\alpha)} - b$. Por ser éste el valor de la asíntota, ningún punto de la curva tendrá ordenada igual a y_0, por consiguiente, el punto de la recta que se halla a esta misma altura no tendrá asociado alguno, sin embargo, el nivel del punto asociado es un dato necesario para calcular la cantidad de puntos cubiertos. Sean (m_{inf}, y_0) las coordenadas del punto de intersección entre p^* y la asíntota. Averigüemos cuál es la posición de $\mathcal{C}$ en el Primal que corresponde a (m_{inf}, y_0). Para esto, vamos a hallar el valor de m_{inf} despejándolo de la ecuación de p^*.

$$p^* : y = ax - b$$

Sustituyendo los valores del punto,

$$y_0 = am_{inf} - b$$

$$\frac{a}{\tan(\alpha)} - b = am_{inf} - b$$

$$m_{inf} = \frac{1}{\tan(\alpha)}$$

Por las propiedades de $\tan(x)$,

$$m_{inf} = \tan(\frac{\pi}{2} - \alpha)$$

Obtenemos así que el punto (m_{inf}, y_0) del Dual se interpreta en el Primal como la posición de $\mathcal{C}$ en la que la semirrecta inferior pasa por el punto (a, b) formando un ángulo α con la prolongación positiva del eje `OY`.

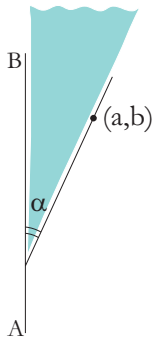


Figura 5.33: Configuración de $\mathcal{C}$ asociada a cualquiera de los puntos de la asíntota horizontal.

Estas son las posiciones de $\mathcal{C}$ que podemos pasar por alto, ya que la semirrecta superior coincide en este caso con la vertical, sin embargo, para garantizar la homogeneidad del algoritmo, consideraremos que cuando un vértice de p^* se encuentre a la altura de la asíntota horizontal de $c(p^*)$, tendrá punto asociado, y que el nivel de éste será cero.

(b) Asíntotas Verticales:
Que un vértice de $\mathcal{A}^*$ pertenezca a una asíntota vertical no es un problema, ya que todo punto de la asíntota vertical va a tener su asociado, siempre y cuando no pertenezca también a la asíntota horizontal. Por curiosidad, sin embargo, veremos cuáles son las posiciones del Primal que dan lugar a estos puntos.

Sea $x = -\frac{1}{\tan(\alpha)}$ la ecuación de la asíntota vertical de la misma curva. Llamemos x_0 al valor $-\frac{1}{\tan(\alpha)}$. Por ser x_0 el valor de la asíntota, ningún punto de la curva tendrá abscisa igual a x_0, lo que se traduce en el Primal en semirrectas superiores con pendiente diferente de x_0. Para ver cuáles son las posiciones de $\mathcal{C}$ en las que la

semirrecta superior tiene pendiente x_0 tenemos en cuenta que por las propiedades de $\tan(x)$, se dan las siguientes igualdades:

$$-\frac{1}{\tan(\alpha)} = -\tan(\frac{\pi}{2} - \alpha) = \tan(-\frac{\pi}{2} + \alpha)$$

Las posiciones (x_0, y) no son otras que aquellas en las que la semirrecta inferior coincide con la vertical, por lo que forman parte de las cuñas que no fueron tenidas en cuenta a la hora de construir la curva y que sabemos podemos pasar por alto, por tener configuración equivalente a la (i). Véase la figura 5.34.

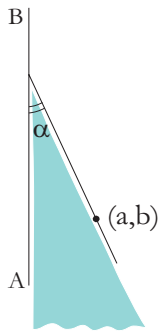


Figura 5.34: Configuración de $\mathcal{C}$ asociada a cualquiera de los puntos de la asíntota vertical.

Las Curvas y el Arreglo

(1) Las curvas asociadas son monótonas en sentido creciente.

(2) Las rectas del arreglo son monótonas en sentido creciente, al tener todas pendientes positivas. Esto se debe a que todos los puntos en el Primal tienen abscisas de signo positivo, y a que las abscisas se transforman en pendientes de rectas al pasar al Dual.

(3) Gracias a la monotonía de curvas y rectas podemos recorrer una recta y su curva asociada en el mismo sentido de manera paralela de modo que

se facilite la localización de los asociados a los vértices. Recordemos que todo vértice y su asociado están a la misma altura. Para insertar una curva se avanzará entonces en sentido creciente por la curva y la recta correspondiente.

(4) Una recta del arreglo y su curva asociada no se intersectan entre sí. Si lo hacieran, el punto de intersección $(m, -n)$, por pertenecer a la recta, correspondería en el Primal a la recta $y = mx + n$, soporte de la semirrecta inferior de $\mathcal{C}$ y, por pertenecer a la curva, correspondería en el Primal a la recta $y = mx+n$, soporte de la semirrecta superior de $\mathcal{C}$. En otras palabras, estaríamos hablando de una cuña de ángulo cero.

(5) Las curvas asociadas no son continuas en la recta real pero sí lo son a cada lado de la asíntota vertical. Se trabajará con las ramas de la curva por separado.

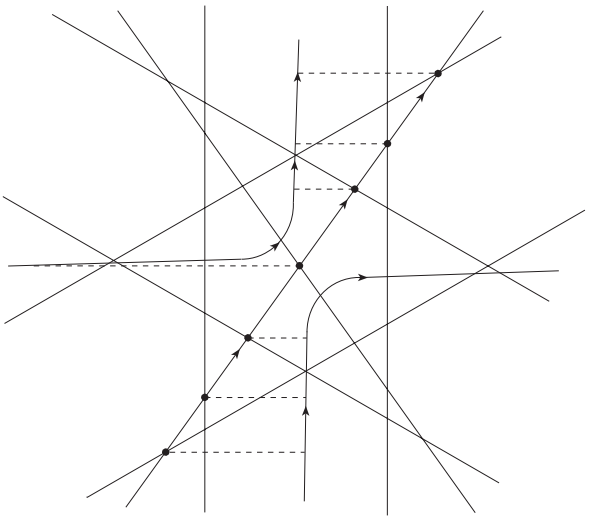


Figura 5.35: Algunos vértices y sus asociados.

Observación 5.4.4. Sólo será necesario trabajar con las ramas a la derecha de la asíntota vertical.

Las ramas que quedan a la izquierda de la asíntota vertical corresponden a semirrectas superiores con pendiente tal que la cuña contiene a la dirección vertical. Como ya sabemos, estas posiciones se reducen a posiciones de configuración (i).

Inserción de las Curvas en el Arreglo

Ya vimos que trabajar con una curva asociada es trabajar con ramas de hipérbola. La intersección entre una rama de hipérbola y cualquier recta del arreglo se da en a lo sumo dos puntos. Gracias al teorema 3.10.1 y al hecho de que las ramas de hipérbola se comportan como arcos pseudoconvexos con respecto al conjunto de rectas podemos concluir que una rama de hipérbola puede insertarse en el arreglo en tiempo lineal. Tenemos entonces el siguiente lema.

Lema 5.4.3. *Las curvas asociadas pueden insertarse en $\mathcal{A}^*$ en tiempo O(n).*

Ya podemos insertar las curvas en el arreglo en tiempo lineal. Veamos ahora los detalles de cómo obtener el número de rectas que hay por debajo de cada punto asociado mientras las curvas se insertan en el arreglo.

Al número de rectas que hay por debajo de un punto asociado x lo llamaremos el nivel de x. El nivel de los puntos asociados se almacenará en sus correspondientes vértices, o sea, los vértices a los que están asociados (por las características del problema, los vértices almacenarán también el número de rectas que inciden en ellos).

Sea entonces $c(p_i^*)$ la curva asociada a p^*. Con la inserción de $c(p_i^*)$ en el arreglo se halla el nivel de todos los puntos asociados que pertenecen a esta curva. Por cada punto de intersección entre $c(p_i^*)$ y el arreglo paramos para ir a p^* y guardar en el vértice correspondiente el nivel del punto asociado por el que $c(p_i^*)$ pasó.

En particular, sea q un punto de intersección entre $c(p_i^*)$ y el arreglo. Al llegar a este punto de intersección vamos a la recta y avanzamos en el mismo sentido que $c(p_i^*)$ mientras los vértices que vayamos encontrando estén por debajo de q o a la

misma altura (estar por debajo de q quiere decir tener menor ordenada).

- En todos estos vértices que estén por debajo de q guardamos como nivel de sus asociados el nivel de la cara de $\mathcal{A}^*$ que $c(p_i^*)$ acaba de visitar.
- En aquel vértice de p^* que esté a la misma altura que q guardamos como nivel de su asociado el nivel de la arista o vértice a que q pertenece. Véase que en este caso el asociado al vértice es precisamente q.

La distinción entre los vértices que están por debajo o a la misma altura que q se debe a que la etiqueta de una cara puede no coincidir con la de alguno de sus vértices adyacentes, como vimos en el capítulo de Técnicas de Implementación.

Una vez actualizados los vértices de p^* seguimos con la inserción de $c(p_i^*)$, o lo que es lo mismo, buscamos el siguiente punto de intersección q entre $c(p_i^*)$ y el arreglo.

Pasos del Algoritmo

Después de hacer la construcción del arreglo $\mathcal{A}^*$ y etiquetar los elementos del mismo con la curva de nivel a que pertencen el trabajo con el arreglo se hace en dos fases, como ya mencionamos anteriormente. Para simplificar la exposición, sin embargo, damos los pasos del algoritmo como si existiera una única fase. El esquema del algoritmo que halla la solución mediante le técnica de paso al Dual es el siguiente.

Algoritmo del Paso al Dual

[1] Convertir los puntos en rectas.

[2] Construir el arreglo de rectas $\mathcal{A}^*$.

[3] Etiquetar vértices, aristas y caras de $\mathcal{A}^*$ con el nivel al que pertenecen.

[4] Etiquetar los vértices con el número de rectas incidentes en el mismo: grado del vértice.

[5] Para toda recta p^* de $\mathcal{A}^*$:

a) construir la curva asociada $c(p^*)$.

b) insertar $c(p^*)$ en $\mathcal{A}^*$.

c) almacenar en cada vértice de p^* el nivel de su asociado.

d) borrar $c(p^*)$ de $\mathcal{A}^*$.

[6] Inicializar k y $k_{\max}$ a cero y $\mathcal{L}$ a la lista vacía.

[7] Para cada vértice r^* y su asociado s^*:

a) Hacer $k = |\text{nivel}(r^*) - \text{nivel}(s^*)| + \text{grado}(r^*)$

b) Si $k > k_{\max}$ hacer $k_{\max} = k$ y vaciar $\mathcal{L}$.

c) Si $k = k_{\max}$ añadir r^* a $\mathcal{L}$.

[10] Devolver $\mathcal{L}$ y $k_{\max}$.

Análisis de la Complejidad. Algoritmo del Paso al Dual.

La complejidad de este algoritmo es $O(n^2)$. La transformación del paso [1] para llevar el conjunto de puntos a un conjunto de rectas puede hacerse en tiempo lineal. La construcción del arreglo de rectas $\mathcal{A}^*$ en el paso [2] lleva tiempo cuadrático utilizando cualquier algoritmo incremental. Los pasos [3] y [4] donde se etiquetan los vértices, aristas y caras del arreglo con el nivel a que pertenecen se hace también en tiempo cuadrático mediante un recorrido del arreglo por las curvas de nivel. El paso [5] se ejecuta en tiempo lineal para cada curva. Al

tratarse de un número de curvas también lineal, este paso tiene complejidad cuadrática. Por último, en el paso [7], los vértices del arreglo pueden visitarse en tiempo cuadrático y en cada uno se realiza una operación en tiempo constante.

Lema 5.4.4. *El algoritmo del Paso al Dual tiene complejidad* $O(n^2)$.

Casos Degenerados

Veamos qué ocurre con las configuraciones del Primal en las que no se cumplen las propiedades de la transformación $\mathcal{T}$.

Puntos Alineados Verticalmente.
Sea $\mathcal{Q}$ un conjunto cualquiera de puntos alineados verticalmente. Sea q la recta que contiene a este conjunto de puntos. La transformación $\mathcal{T}$ no está definida para q, por ser esta vertical; luego, entre $\mathcal{Q}$ y q no se da la propiedad de incidencia de $\mathcal{T}$.

Si el conjunto $\mathcal{Q}$ pertenece a $\overline{AB}$, podrá aparecer en semirrectas verticales, en cuyo caso las posiciones en cuestión pueden ser pasadas por alto, al poderse transformar en posiciones de configuración (i). Si el conjunto $\mathcal{Q}$ no pertenece a $\overline{AB}$, no puede aparecer como conjunto en ninguna semirrecta para ninguna posición de $\mathcal{C}$, luego, que no se preserven las propiedades de incidencia entre estos puntos y la recta que los contiene no tiene ya consecuencias, puesto que esta recta vertical no tiene significado alguno para el problema (en este caso el conjunto $\mathcal{Q}$ pasará al Dual como un conjunto de rectas paralelas entre sí, lo que no tiene inconveniente alguno a la hora de construir el arreglo).

5.5 Resultados

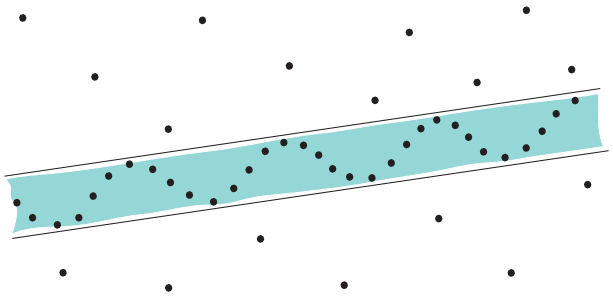
Con todo lo visto hasta ahora tenemos entonces el siguiente teorema:

Teorema 5.5.1. *Dados en plano un segmento, un conjunto de n puntos y una cuña de ángulo fijo, la posición en el segmento donde la cuña cubre la mayor cantidad de puntos puede ser calculada en tiempo* $O(n^2)$.

5.6 Extensiones del Algoritmo

De los tres procedimientos que vimos para trabajar con las configuraciones cuadráticas, el del Paso al Dual es el que tiene mejor complejidad. Este procedimiento puede utilizarse también para resolver el problema en el que la cuña ha sido sustituida por una banda, siendo una banda la región del plano comprendida entre dos rectas paralelas.

Problema 6. *Dados un conjunto de n puntos en el plano y una banda de ancho fijo, hallar la posición de la banda de manera que cubra la mayor cantidad de puntos.*



Al igual que en el caso de la cuña, es posible discretizar el conjunto solución. En este caso, cualquier solución puede llevarse a una configuración en la que una de las rectas de la banda pasa por al menos dos puntos. Estas rectas estarán localizadas en el Dual en los vértices del arreglo. Para localizar la otra recta de las bandas se inserta igualmente una curva por cada recta del arreglo.

Bibliografía

[1] L. Guibas B. Chazelle and D. Lee. The Power of Geometric Duality. *BIT, 25(1)*, pages 76–90, 1985.

[2] I. J. Balaban. An Optimal Algorithm for finding Segments Intersections. *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 211–219, 1995.

[3] B. Chazelle and D. T. Lee. On a Circle Placement Problem. *Computing*, 36:1–16, 1986.

[4] J. E. Goodman and J. O'Rourke. *Handbook of Discrete and Computactional Geometry*. CRC Press, 1997.

[5] R. Seidel H. Edelsbrunner and M. Sharir. On the Zone Theorem for Hyperplane Arrangements. *SIAM J. Comp. 22(2)*, pages 418–429, 1993.

[6] S. Hart and M. Sharir. Nonlinearity of Davenport-Schinzel Sequences and of generalized Path Compression Schemes. *Combinatorica 6*, pages 151–177, 1986.

[7] R. Seidel J. O'Rourke and H. Edelsbrunner. Constructing Arrangements of Lines and Hyperplanes with Applications. *SIAM J. Comput., vol. 15*, pages 341–363, 1986.

[8] D. E. Knuth. *Sorting and Searching, in The Art of Computer Programming*, volume 3. Addison Wesley, Reading, MA., 1973.

[9] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, 1992.

[10] M. Sharir P. Agarwal and P. Shor. Sharp upper and lower Bounds for the Length of general Davenport-Schinzel Sequences. *J. Combin. Theory, Ser. A 52*, pages 228–274, 1989.

[11] J. O' Rourke. *Computational Geometry in C.* Cambridge University Press, 1994.

[12] M. Sharir. Dagstuhl Seminar 03121, at Schloss Dagstuhl. International Conference and Research Center for Computer Science, March 2003.

[13] M. Sharir and P. K. Agarwal. *Davenport-Schinzel Sequences and their Geometric Applications.* Cambridge University Press, 1995.

[14] I. Streinu. Non-stretchable pseudo-visibility graphs. *Proc. 11th Canadian Conference of Computational Geometry CCCG'99, Vancouver, Aug.*, 1999.

[15] R. Tarjan. Efficiency of a good but not linear Set-union Algorithm. *Journal of Association for Computing Machinery 22*, pages 215–225, 1975.